UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO FIN DE GRADO

DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN PARA LA GESTIÓN DEL CORTAFUEGOS DE ANDROID

Autor: Antonio Requena López

Tutor: Sergio Pastrana Portillo

Leganés, Marzo de 2016

Título: DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN PARA LA GESTIÓN DEL CORTAFUEGOS DE ANDROID

Autor: Antonio Requena López

Director: Dr. Sergio Pastrana Portillo

EL TRIBUNAL

Presidente:

Vocal:

Secretario:

Realizado el acto de defensa y lectura del Trabajo Fin de Grado el día 8 de Marzo de 2016 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle una aplicación de:

Agradecimientos

Este trabajo supone el cierre de una de las etapas más importantes de mi vida, si bien es cierto que por un lado ha sido muy costoso llegar hasta aquí, lo único que guardo de mi etapa como estudiante en la universidad son buenos recuerdos. Muchas personas han estado a mi lado durante esta etapa y creo que se merecen una mención especial en este pequeño apartado.

En primer lugar agradecérselo a mis padres por darme todas las facilidades posibles y apremiarme en todo momento a seguir estudiando y aprendiendo para el día de mañana. A mi novia, Vanesa, por estar a mi lado durante casi toda la carrera apoyándome para poder llegar hasta aquí.

A mis compañeros de la universidad como Pedro, Luis, David o Noelia, con los que he compartido incontables horas en la biblioteca preparándonos para aquellos exámenes tan terribles

A mis amigos de toda la vida Alberto, Óscar y Víctor con los que siempre tomaba unas cervezas después de clase y compartía con ellos todo lo vivido durante la semana

A mi tutor de este Trabajo de Fin de Grado, Sergio, que aún estando hasta arriba de trabajo siempre tuvo un momento para orientarme y ayudarme durante el desarrollo de este trabajo.

A todas las personas que componen (o componían) el área de Espacio Estudiantes de Getafe, y en especial a Josué, que fue mi responsable durante varios años en el que fue uno de mis primeros trabajos enfocado a la informática.

A todos los profesores que de un modo u otro influyeron en mi aprendizaje y formación durante estos años, de muchos de ellos guardo buenos recuerdos y fueron de muchísima ayuda.

Y finalmente a ti, que estás empleando un poco de tu tiempo en leer estas líneas y este trabajo que tantas horas y esfuerzo han costado.

A todos, gracias,

"Podría parecer que hemos llegado a los límites alcanzables por la tecnología informática, aunque uno debe ser prudente con estas afirmaciones, pues tienden a sonar bastante tontas en cinco años"

John Von Neumann

Resumen

Hoy en día vivimos en una sociedad que hace uso con mucha frecuencia de las nuevas tecnologías, las cuales evolucionan aceleradamente con al paso del tiempo. Un perfecto ejemplo de esto es el auge del uso de dispositivos inteligentes como los smartphones o tablets.

Desde el móvil podemos consultar nuestra cuenta bancaria, comunicarnos con amigos o familiares, interactuar con las redes sociales, hacer fotos o incluso realizar compras a través de portales web o aplicaciones. El teléfono móvil cuenta con un amplio abanico de características que nos hacen la vida más fácil y nos proporcionan más comodidad en nuestra vida diaria

Sin embargo, la amplia funcionalidad ofrecida por los teléfonos inteligentes requiere del almacenamiento de cierta información privada como datos de localización, datos bancarios, fotografías, etc. Si esta información cae en las manos equivocadas podría incurrir en un daño muy elevado al usuario.

Se hace por lo tanto necesario el poder dotar de herramientas que permitan asegurar la confidencialidad, integridad y disponibilidad de los datos, de igual manera que se ha estado trabajando en ordenadores convencionales. De esa idea, surge este proyecto, cuya principal intención es la de otorgar al dispositivo una capa de seguridad adicional que dificulte el acceso a los datos por parte de un posible acceso ilegítimo al teléfono y a los datos que este almacena, mediante la configuración de un cortafuegos de aplicaciones.

Abstract

Nowadays, we live in a society which is highly dependable on new technologies, which involve rapidly through time. A perfect example of such evolution is the increased use of smart devices such as smartphones or tablets.

Using a Smartphone, we can check our bank account, communicate with friends or our family, interact with social networks, take pictures or even perform buy through web portals or applications. The mobile phone has a wide range of features that make us the life easier and give us more comfort in our daily lives.

However, the useful functionality provided by these smart devices requires the storage o personal and private information, such as location data, banking or images and pictures.. If such information is compromised, users can be at a high danger.

It is this required to provide the new technologies with security tools aiming at provide confidentiality, privacy and integrity to the data, same as those intended for personal computers. This is the main motivation of this project, whose main objective is providing at the device a new additional layer of security (i.e. a firewall) that prevents the access at the data storage inside the mobile of a possible illegal access to telephone and data stores.

Índice

Índice de Ilustraciones

Índice de tablas

CAPÍTULO 1

Introducción

En este capítulo se realiza una aproximación de las facetas más importantes de este proyecto, como por ejemplo la motivación inicial, los principales problemas que soluciona y los objetivos que la plataforma debe de cumplir para solucionar dichos problemas. Adicionalmente se incluyen unas tablas de términos y de abreviaturas que se pueden encontrar dentro del documento.

1.1 Área del proyecto

El mundo de la tecnología avanza a pasos agigantados, se trata de un área con muchas innovaciones y nuevas evoluciones en muy poco tiempo. Cada poco tiempo surge una nueva tecnología que revoluciona el panorama de la informática y que cambia nuestra forma de interactuar con ella en menor o mayor medida.

Dentro de estos avances se encuentran los "*smartphones*", una gama de dispositivos que están dotados de software capaz de adaptarse a las necesidades de los usuarios realizando tareas como envíos de emails, localización de la posición, compras, navegar por internet, etc. Además de realizar las tareas que ya realizaban de por sí los teléfonos móviles.

Los teléfonos inteligentes han supuesto una autentica revolución en el mercado, no hay más que ver las estadísticas para comprender que nos encontramos ante un fenómeno que supone al mercado una cantidad altísima de ingresos[1]

Ilustración 1. Ventas mundiales de smartphones

Junto con la venta de los dispositivos, el mercado de las aplicaciones desarrolladas para correr en para ellos también ha experimentado una gran subida en lo que se refiere a la oferta y a la demanda, por lo que supone una gran oportunidad para programadores con buenas ideas.

1.2 Motivación

La versatilidad que ofrecen los *smartphones* es muy útil, pero para que estos dispositivos puedan ofrecer sus características, es necesario que almacenen cierta información privada por parte del propietario del teléfono. Esta información en la mayoría de los casos, son claves, conversaciones privadas o datos bancarios.

Almacenar toda esa información dentro del dispositivo es muy peligroso, ya que en el caso de perderlo o de sufrir un robo, esa información sería conveniente que no

[1] http://www.telesemana.com/blog/2015/08/25/estadisticas-mercado-de-smartphones-a-nivel-mundial/

viera la luz puesto que, además de ser privada, puede permitir una suplantación de identidad por parte de otra persona y podría llegar a producir un daño difícil de reparar para el propietario del teléfono

Ante la perspectiva que presenta esta posibilidad, la visión que se presenta es clara. Se debe de integrar una herramienta que impida el acceso a las aplicaciones, y a los datos que estos almacenan, dotando a los teléfonos de una capa extra de seguridad que nos permita disfrutar de todas las ventajas que nos ofrecen con el menor riesgo posible.

1.3 Problema

El problema que se tratará de resolver mediante este proyecto es el de proveer de una capa adicional de seguridad dentro de los teléfonos, para que en caso de robo, pérdida o de instalación de aplicaciones potencialmente maliciosas, los datos almacenados en el teléfono no se encuentren directamente expuestos.

Para poder realizar esta tarea se desarrollará una aplicación para la gestión del cortafuegos integrado dentro del sistema operativo Android. La aplicación desarrollada se encargará de interactuar y de establecer unas reglas que sean impuestas por el usuario, como las aplicaciones que se bloquean cuando se está cargando la batería o cuando está el wifi activado.

A pesar de que el cortafuegos es una herramienta muy poco flexible, la aplicación está diseñada de manera modular, es decir, permite añadir nuevas funciones de configuración sobre el cortafuegos, de manera que dado múltiples casuísticas puede mantenerse correctamente configurado cumpliendo con los deseos del usuario.

1.4 Objetivos del Trabajo

El objetivo principal de este Trabajo de Fin de Grado es el de **desarrollar y documentar una aplicación para gestionar y configurar el cortafuegos de Android**. Se consideran los siguientes dos subobjetivos:

- *Estudio previo*. Analizar qué soluciones existentes permiten configurar y bloquear aplicaciones en sistemas operativos Android.
- *Diseño modular*. Elaborar un plan de desarrollo de la manera más óptima posible, como si se tratara de un encargo real realizado por un cliente al que se le está dando un servicio pero que puede sufrir modificaciones y tener módulos adicionales en el futuro.

Para cumplir con este objetivo se redactaran unos requisitos, se establecerá un modelo arquitectónico y se realizarán unas pruebas que garanticen el correcto funcionamiento de la aplicación.

1.5 Contenido de la memoria

En este documento se detallan todos los aspectos que son necesarios para el desarrollo de este proyecto. Los puntos que componen la memoria son:

- Capítulo 1: Introducción: Incluye una breve descripción del documento, se explica el problema que se quiere resolver y como se va a resolver, además de otra información adicional como términos y abreviaturas.

- Capítulo 2: Estado del arte: Se explica en qué estado se encuentra las tecnologías usadas, sus características, sus datos de ventas, etc. También se detalla de una manera más específica en qué consiste el funcionamiento de la aplicación.

- Capítulo 3: Análisis del sistema: En este punto se detallan los requisitos que definen la aplicación y una serie de casos de uso.

- Capítulo 4: Diseño de la aplicación: Este apartado define la arquitectura del sistema, los diagramas de secuencia y se detalla la implementación del sistema.

- Capítulo 5: Pruebas: Esta entrada define una serie de pruebas que se encargan de verificar el correcto funcionamiento de la aplicación, demostrando que cumple con su cometido.

- Capítulo 6: Planificación y presupuesto: En este punto se muestra una planificación inicial del trabaja, una planificación final con todos los datos reales acerca del tiempo que se ha invertido en el desarrollo del proyecto y un presupuesto que refleja el coste de su realización.

- Capítulo 7: Conclusiones: Se presentan una serie de conclusiones alcanzadas tras la realización del proyecto.

De manera adicional, se añade al final del documento un listado con las referencias usadas para la elaboración de la memoria y un anexo con un manual de usuario para el uso de la aplicación.

CAPÍTULO 2

Estado del Arte

Este capítulo sirve como una base teórica sobre la que se construye este Trabajo de Fin de Grado. Se comienza hablando sobre el Sistema Operativo Android, su arquitectura y su funcionamiento interno. Seguidamente se explica el funcionamiento del cortafuegos así como la idea básica que se tenía la comienzo del proyecto y como a evolucionado este desde el inicio de la investigación.

2.1 El Sistema Operativo Android

En Octubre del año 2003 se funda en Palo Alto Android Inc. La idea es desarrollar un sistema operativo para móviles basado en el Sistema Operativo Linux. En Julio del año 2005, Google compra esta compañía y en Noviembre de 2007 se funda la Open Handset Alliance [2](OHA), la cual trata de una alianza comercial de 84 compañías que se dedica a desarrollar estándares abiertos.

Ilustración 2. Compañías que componen la OHA

El mismo día que se funda la OHA, se anuncia la primera versión de Android "*Apple Kie"*. Los terminales con android no estarán disponibles hasta el año 2008, el primero fue el "*HTC Dream"*.

A modo de curiosidad, tanto el nombre del Sistema Operativo, como los Smarphones propios de Google (*Nexus)* poseen un nombre extraído de la novela *"¿Sueñan los androides con ovejas eléctricas?"* de Philip K. Dick, novela que posteriormente sería adaptada al cine con el título de "*Blade Runner"*.

Ilustración 3. Logo de los dispositivos Nexus

[2] http://busyprogrammer.com/introduction-to-android/

2.1.1 Características generales de Android

Cada cierto tiempo, y desde el lanzamiento de Android, el Sistema Operativo recibe nuevas versiones que son denominadas como "Actualizaciones", con cada actualización se le dota al Sistema Operativo de mejoras y de nuevas características.

Entre las principales características que posee Android, se encuentran las siguientes[3]:

- Diseño de dispositivo: La plataforma es adaptable a pantallas de mayor resolución, VGA y bibliotecas de gráficos 2D y 3D.

- Almacenamiento: Para almacenar datos Android usa una base de datos liviana, SQLite.

- Conectividad: Android soporta las siguientes tipologías de conectividad: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, HSDPA, HSPA+, NFC y WiMAX, GPRS, UMTS y HSDPA+.

- Mensajería: El sistema soporta SMS y MMS como mensajería de texto, también soporta *Android Cloud to Device Messaging* (C2DM), como parte del servicio que ofrece Android de Push Messaging.

- Navegador Web: Android posee un navegador web que está basado en un motor de renderizado de código abierto WebKit junto con el motor JavaScript V8 de Google Chrome.

- Soporte de Java: Ya que la mayoría de las aplicaciones son escritas en Java y la base del Sistema Operativo es Linux, para la ejecución de las aplicaciones Android posee una máquina virtual Dalvik especializada y diseñada específicamente para Android y dispositivos móviles con batería y con, poca memoria y un procesador limitado.

- Soporte Multimedia: Android posee soporte para varios formatos de archivos multimedia, como: WebM, H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF y BMP.

- Soporte para Streaming: El sistema soporta el Streaming RTP/RTPS y la descarga progresiva de HTML. Soporta también el Adobe Flash Streaming mediante Adobe Flash Player.

[3] http://androidos.readthedocs.org/en/latest/data/caracteristicas/

- Soporte para Hardware adicional: Android soporta cámaras de fotos, de video, pantallas táctiles, GPS, acelerómetro, giroscopios, magnetómetros, sensores de proximidad, sensores de presión, sensores de luz, gamepad, termómetro y aceleración por GPU 2D y 3D.

- Entorno de desarrollo: Dentro de Android se incluye un emulador de dispositivos, herramientas para depuración de memoria y de análisis del rendimiento del software. Inicialmente, el entorno usado para desarrollar las aplicaciones era Eclipse con el plugin de "*Herramienas de Desarrollo de Android"* (ADT) , actualmente el entorno de desarrollo oficial de las aplicaciones es Android Studio.

- Google Play: Se trata del catálogo de aplicaciones (gratuitas o de pago) en el que se pueden descargar las aplicaciones e instalarlas en el dispositivo sin la necesidad de tener un ordenador de por medio.

- Multi-táctil: Android posee soporte para pantallas capacitivas con soporte multi-táctil. La funcionalidad fue inicialmente desactivada a nivel de Kernel, posiblemente para evitar infringir patentes de otras compañías.

- Bluetooth: En la versión 1.5 se agregó el soporte para A2DF y AVRCP. El envío de archivos (OPP) y la exploración del directorio telefónico se agregaron en la versión 2.0 Finalmente, en la versión 2.2 se añadió el marcado por voz y el envío de contactos entre teléfonos.

- Videollamada: Haciendo uso de la aplicación *"Hangouts"*, Android posee soporte parala realización de videollamadas desde la versión *"HoneyComb"*.

- Multitarea: Android posee la opción de la multitarea, es decir, permite que las aplicaciones que no se ejecutan en primer 'plano, sigan recibiendo ciclos de reloj y permiten su ejecuciñon en segundo plano.

- Características basadas en voz: Desde la versión inicial del Sistema Operativo, se encuentra disponible la opción de realizar búsquedas por Google a través de la voz del usuario como elemento de entrada.

- Tethering: Android soporta la opción del *"Tethering"*, es decir, permite al teléfono ser usado como un punto de acceso inalámbrico. Inicialmente esta opción se realizaba a través de aplicaciones, pero a partir de la versión 2.2 fue incluida dentro del sistema operativo. Para que un ordenador pueda

hacer uso de la conexión 3G del dispositivo podría ser necesario la instalación de software adicional.

2.1.2 Versiones y características

Desde el lanzamiento de la primera versión de Android allá por el año 2008, Google ha ido lanzando de manera periódica actualizaciones que dotaban a los dispositivos de nuevas características y mejoras de rendimiento y consumo.

Inicialmente, Android fue pensado y diseñado para usarlo en *"smartphones"*, pero con la aparición de las *"tablets"* en el mundo de la electrónica Google se vio obligado a realizar distinciones en sus versiones. Desde la versión 1.0 A la 2.XX las actualizaciones estaban pensadas para teléfonos, y las versiones de enumeración 3.XX estaban dedicadas para las tablets. A partir de la 4.XX las actualizaciones salían de manera conjunta tanto para teléfonos como tabletas.

A continuación se enumeran las distintas versiones[4] y las principales características que estas poseen:

Versión	1.0	**Nombre**	Apple Pie	**Logo**
Características	Inclusión del Android Market, el navergador web, soporte para la cámara, acceso a servidores de correo, sincronización con Gmail, Google Contacts, Calendar y Maps, soporte para Wi-Fi y Bluetooth.			 Ilustración 4. Android Apple Pie
Fecha de Lanzamiento	23 de Septiembre de 2008			

Tabla 1. Android 1.0

Versión	1.1	**Nombre**	Banana Bread	**Logo**
Características	Mejoras en Google Maps, Posibilidad de guardar archivos adjuntos en los mensajes, soporte para marquesina en diseños de sistemas.			 Ilustración 5. Android Banana Bread
Fecha de Lanzamiento	9 de Febrero de 2009			

Tabla 2. Android 1.1

[4] http://rootear.com/android/conoce-las-diferencias-entre-las-distintas-versiones-de-android
http://www.xatakamovil.com/sistemas-operativos/de-cupcake-a-marshmallow-asi-han-sido-las-versiones-de-android-a-lo-largo-de-su-historia

Versión	1.5	Nombre	Cupcake	Logo
Características	Permite subir videos a Youtube e imágenes a Picasa, soporte para widgets y teclados virtuales, agregada opción de auto rotación de pantalla, añadidas las características de copiar y pegar, agregada la animación de inicio oficial, marcas de fecha y hora para el registro de eventos.			 Ilustración 6. Android Cupcake
Fecha de Lanzamiento	30 de Abril de 2009			

Tabla 3. Android 1.5

Versión	1.6	Nombre	Donut	Logo
Características	Mejora en la búsqueda por texto y voz, mejor integración de galería, cámara y videocámara, motor multi-lenguaje de síntesis de habla, soporte para la selección múltiple en la galería, mejoras de velocidad en búsqueda y aplicaciones de cámara.			 Ilustración 7. Android Donut
Fecha de Lanzamiento	15 de Septiembre de 2009			

Tabla 4. Android 1.6

Versión	2.0 / 2.1	Nombre	Eclair	Logo
Características	Posibilidad de vincular varias cuentas de gmail al dispositivo, soporte de intercambio de correo, soporte bluetooth 2.1, nueva interfaz de usuario para el navegador, optimización en velocidad de hardware y GUI renovada, mejorado Google Maps, posibilidad de establecer fondos de pantalla animados.			 Ilustración 8. Android Eclair
Fecha de Lanzamiento	26 de octubre de 2009			

Tabla 5. Android 2.0 / 2.1

Versión	2.2	Nombre	Froyo	Logo
Características	Optimización general sobre memoria, rendimiento y velocidad de aplicaciones. Integración del motor JavaScript v8, uso del *"Tethering"*, marcación por voz, instalación de aplicaciones en memoria expandible.			 Ilustración 9. Android Froyo
Fecha de Lanzamiento	20 de Mayo de 2010			

Tabla 6. Android 2.2

Versión	2.3.X	Nombre	Gingerbread	Logo
Características	Soporte para pantallas extra grandes, soporte para SIP y VoIP, soporte para NFC, nuevo gestor de descargas, mejoras en la administración de la energía, coporte para giroscópio y barómetro, soporte para múltiples cámaras.			 Ilustración 10. Android Gingerbread
Fecha de Lanzamiento	06 de Diciembre de 2010			

Tabla 7. Android 2.3.X

Versión	3. X	Nombre	Honeycomb	Logo
Características	Soporte optimizado para tablets, agregada barra de sistema, teclado rediseñado, multitarea simplificada, nueva interfaz de correo, aceleración de hardware, habilidad para cifrar datos del usuario.			 Ilustración 11. Android Honeycomb
Fecha de Lanzamiento	22 de Febrero de 2011			

Tabla 8. Android 3.X

Versión	4.0.X	Nombre	Ice Cream Sandwich	Logo
Características	Botones en la pantalla, separación de widgets, facilidad para crear carpetas, lanzador personalizable, buzón de voz mejorado, corrector ortográfico, captura de pantalla integrada, editor de fotos, grabación de video a 1080P, nuevo diseño de la galería, desbloqueo facial.			 Ilustración 12. Android Ice Cream Sandwich
Fecha de Lanzamiento	19 de Octubre de 2011			

Tabla 9. Android 4.0.X

Versión	4.1 – 4.3	Nombre	Jelly Bean	Logo
Características	Soporte de bluetooth de baja energía, de OPENGL ES y de resolución 4K, inclusión de nuevos idiomas, sistema de localización Wi-Fi, función de autocompletado en el *"Dial Pad"*, sistema de logging mejoras de seguridad.			
Fecha de Lanzamiento	27 de Junio de 2012			

Ilustración 13. Android Jelly Bean

Tabla 10. Android 4.1

Versión	4.4.X	Nombre	KitKat	Logo
Características	Nuevo diseño de la interfaz, nueva gestión de optimización en dispositivos con especificaciones técnicas comedidas, posibilidad de impresión mediante Wi-Fi, nuevos widgets, arreglos en la conexión a datos.			
Fecha de Lanzamiento	31 de Octubre de 2013			

Ilustración 14. Android Kit Kat

Tabla 11. Android 4.4.X

Versión	5.0	Nombre	Lollipop	Logo
Características	Nueva interfaz y nuevos patrones de diseño denominados Material Design, nueva rececpción de las notificaciones, modo de ahorro de batería, soporte para GPUs de 64 bits, soporte para múltiples tarjetas SIM, mejoras de estabilidad y rendimiento.			
Fecha de Lanzamiento	3 de Noviembre de 2014			

Ilustración 15. Android Lollipop

Tabla 12. Android 5.0

Versión	6.0	Nombre	Marshmallow	Logo
Características	Inclusión de un administrador de permisos, soporte para el pago mediante NFC, soporte para uso de Huella dactilar, soporte para tarjetas SD, soporte para restauraciones copias de seguridad completas			
Fecha de Lanzamiento	5 de Octubre de 2015			

Ilustración 16. Android Marshmallow

Tabla 13. Android 6.0

2.1.3 Comparación de versiones

Uno de los principales problemas a los que tiene que hacer frente Android es la actualización de los dispositivos móviles en un periodo de tiempo más o menos aceptable. Debido a que cada fabricante tiene su propia capa de personalización sobre el Sistema Operativo, no todos los dispositivos se actualizan en el mismo periodo de tiempo.

En la siguiente Figura, se puede apreciar una gráfica donde se comparan las versiones de Android que actualmente se encuentran más en uso. Estos datos están recogidos a principios de Febrero del año 2016[5]

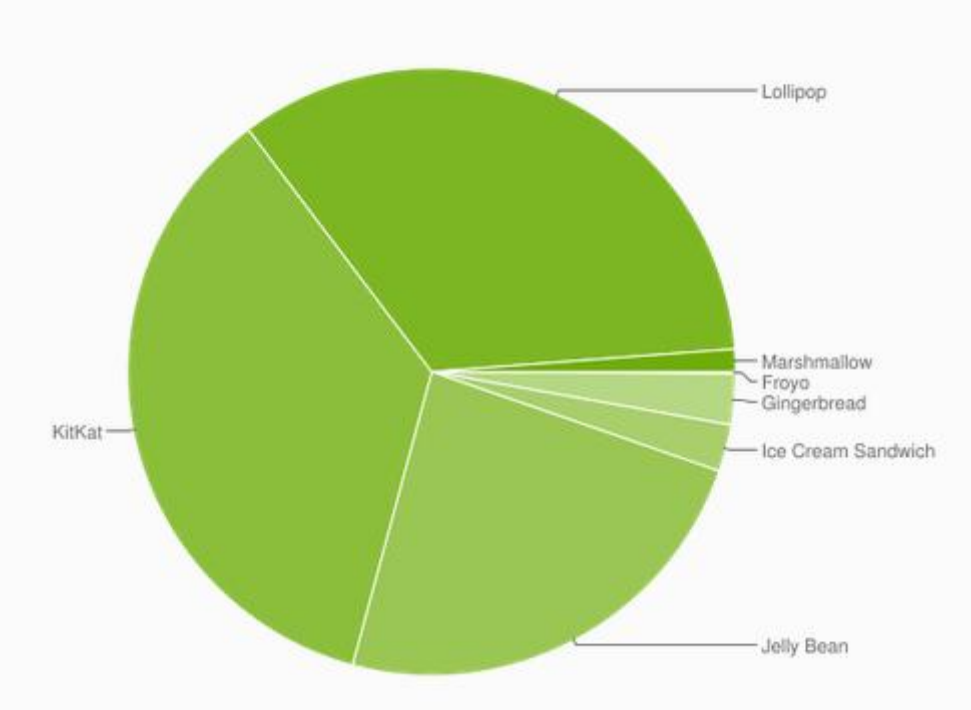


Ilustración 17. Comparación de versiones Febrero 2016

Versión	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 – 2.3.7	Gingerbread	10	2.7%
4.0.3 – 4.0.4	Ice Cream Sandwich	15	2.5%
4.1.X	Jelly bean	16	8.8%
4.2.X	Jelly Bean	17	11.7%
4.3	Jelly Bean	18	3.4%
4.4	Kit Kat	19	35.5%
5.0	Lollipop	21	17.0%
5.1	Lollipop	22	17.1%

[5] http://www.xatakandroid.com/mercado/android-marshmallow-sigue-creciendo-timidamente-solo-el-0-7-de-los-dispositivos-han-actualizadov

6.0	Marshmallow	23	1.2%

Tabla 14. Comparativa de versiones

La versión actual que más se encuentra en uso es KitKat, la cual salió a la luz en Octubre del 2013, poco a poco la cuota de Lollipop crece, estando cerca de ser la más usada, y la nueva versión, Marshmallow se encuentra en apenas el 1.2% de los dispositivos.

Aquí es donde se aprecia el principal problema de Android, versiones como Lollipop que apareció en Noviembre de 2014, no es aún la más usada, y una versión como marshmallow que apareciçon en Octubre de 2015, apenas se encuentra en un 1% de los dispositivos.

Para dejar aún más constancia de este problema, en la comparativa de Febrero del 2015[6], vemos que apenas 3 meses después de la salida de Lollipop, esta versión apenas se encontraba extendida.

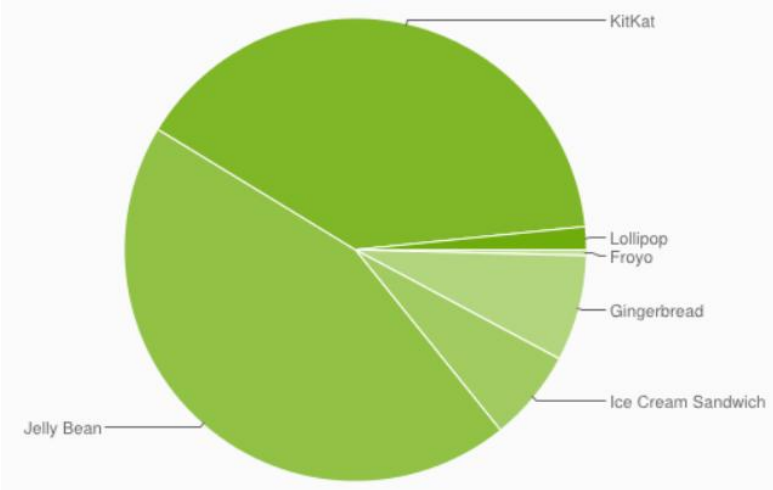


Ilustración 18. Comparación de versiones Febrero 2015

[6] http://www.xatakandroid.com/mercado/mas-del-40-de-los-dispositivos-android-llevan-kitkat-y-solo-el-3-3-estan-actualizados-a-lollipop

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	7.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	6.4%
4.1.x	Jelly Bean	16	18.4%
4.2.x		17	19.8%
4.3		18	6.3%
4.4	KitKat	19	39.7%
5.0	Lollipop	21	1.6%

Ilustración 19. Tabla de versiones en Febrero 2015

Otro punto que llama la atención es que versiones como Gingerbread o Froyo (Aunque esta última en un margen mucho más pequeño), siguen teniendo existencia dentro de la cuota de mercado de Android, por lo que queda constancia de que las actualizaciones dentro de Android son muy lentas, llevan su tiempo y las versiones obsoletas tardan mucho en desaparecer.

2.1.4 Arquitectura

Como todo sistema operativo, Android posee su propia arquitectura[7], la cual detallaremos a continuación.

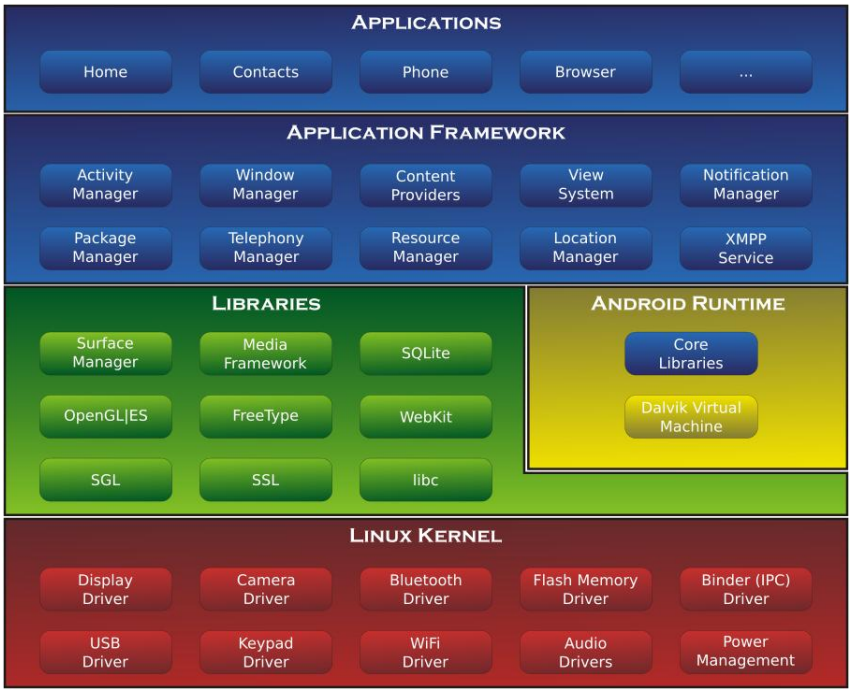


Ilustración 20. Arquitectura de Android

[7] http://lslutnfra.blogspot.com.es/2013/01/arquitectura-de-android-os-curso.html

La arquitectura de Android está basada en un Kernel de Linux, con middleware, librerías y APIs escritas en C, y aplicaciones que son ejecutadas en un "*framework*" de aplicaciones que incluye librerías compatibles con Java.

Para poder optimizar al máximo los recursos del dispositivo, ya que la memoria el procesador y el almacenamiento son escasos, Android usa una máquina virtual Dalvik en lugar de una máquina virtual con Java.

El desglose de la arquitectura en capas hace que la separación de las tareas sea más sencilla, pero implica que cada capa depende de la anterior para poder realizar su tarea. Este tipo de arquitectura es comúnmente denominada "*Pila".* A continuación se detalla la función de cada una de las capas:

- Kernel de Linux: El núcleo de Android está basado en un kernel de Linux, concretamente en la versión 2.6, esta capa aporta los servicios base del sistema como pudeen ser el modelo de controladores, la gestión de procesos, gestión de memoria, la pila de seguridad o de red. El desarrollador no accede a esta.

- Librerías: Android dispone de un conjunto de bibliotecas en C y C++ usados por varios componentes del sistema. Las librerías proporcionan funcionalidades que se repiten frecuentemente como, por ejemplo, el motor gráfico, el cifrado de las comunicaciones o la renderización de fuentes.

- Runtime de Android: Android posee un conjunto de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las librerías.

  Dentro de esta capa se encuentra también la máquina virtual Dalvik. Cada vez que se ejecuta una aplicación en Android esta tiene su propio proceso y su propia instancia dentro de la máquina Dalvik. Dalvik puede ejecutar varios procesos a la vez ejecutando los archivos *".dex"*

- Marco de trabajo de las aplicaciones: Representan fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. En este nivel se encuentran representadas las APIS de las cuales hacen uso los desarrolladores para poder programar las aplicaciones.

- Aplicaciones: En esta capa se encuentran aglomeradas las aplicaciones que están instaladas en el dispositivo. Además de las aplicaciones que ya vienen instaladas de serie en el dispositivo, el usuario podrá instalar las aplicaciones que él desee.

2.2 Comparativa de Sistemas Operativos

Actualmente, y con la importancia que han ido adquiriendo los *smartphones* en el mundo de la tecnología, han ido apareciendo más sistemas operativos y más variedad de dispositivos móviles dentro del mercado. Cada Sistema operativo ofrece sus propias ventajas y sus propios defectos de cara a los usuarios.

Aparte de Android, los principales sistemas operativos para *smartphones* son iOS de Apple, Windows Phone de Microsoft y Blackberry OS desarrollado por BlackBerry. Además de estos, existen más sistemas operativos como Symbian desarrollado por Nokia, pero se han descartado por que o bien no tienen apenas cuota de mercado, o bien se encuentran en decadencia[8].

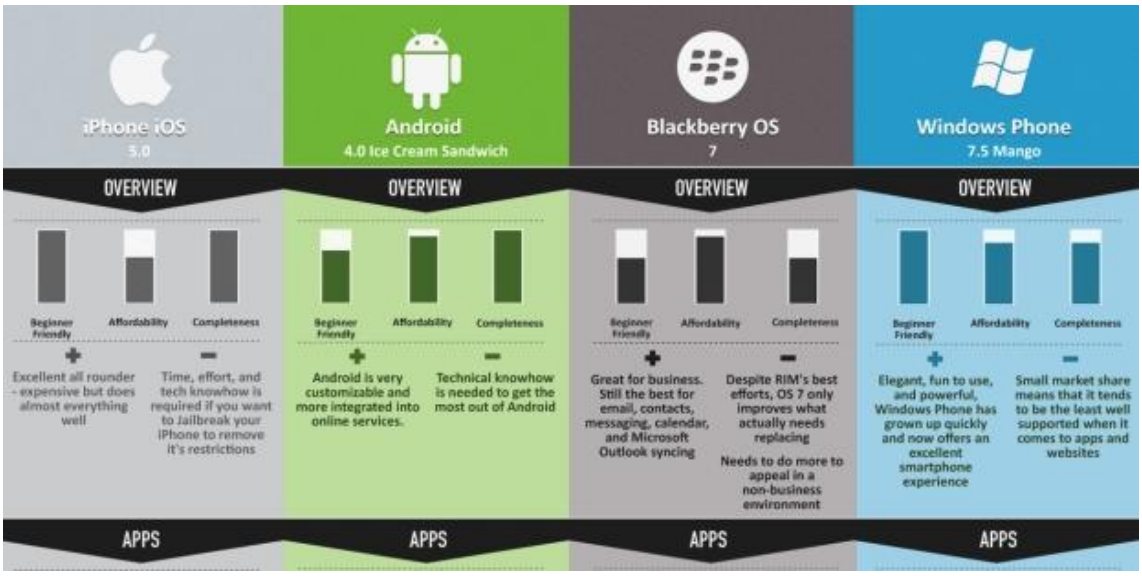


Ilustración 21. Comparativa de los Sistemas Operativos

- iOS: El sistema operativo de Apple es, sin lugar a dudas, un sistema excelente, con un gran rendimiento y una brillante fluidez, su principal inconveniente es que sus dispositivos son exageradamente caros. Posee un amplio abanico de aplicaciones dotadas de una gran usabilidad y una interfaz elegante.

  iOS posee un sistema de almacenamiento en la nube a través de su plataforma iCloud. En ella el usuario puede almacenar datos y copias de seguridad de manera que siempre que posea conexión a internet podrá restaurar la copia de seguridad o acceder a algunos de los datos (fotos, videos, música) que haya almacenado en su interior.

[8] http://www.hardwareperu.com/windows-phone-vs-android/

Los teléfonos de Apple poseen un hardware de gran calidad, pero a diferencia de Android no ofrece variedad en dispositivos, siendo todos los que ofertan dispositivos de gama alta y de un precio elevado.

- Android: La principal característica de Android es su variedad, ya que al ser un sistema apadrinado por varios fabricantes, se goza de una amplia gama de dispositivos que se ajustan a lo que demanda el usuario. Por otra parte el precio de estos dispositivos no son muy elevados y ofrecen una relación calidad-precio envidiable.

  Al igual que iOS, Android posee también un amplio abanico de aplicaciones, y permite sincronizar los datos de sus *smartphones* con sus distintas herramientas como GMail, Google Docs o Google Drive.

  Android ha ido mejorando poco a poco su interfaz y su usabilidad llegando a un punto en el que no tiene nada que envidiar al resto de fabricantes, lo único que se le podría reprochar es intentar sacar mayor rendimiento al hardware, algo que sin duda es muy difícil, ya que abarca un amplio abanico de dispositivos y cada uno de fabricantes distintos.

  Según un estudio de Symantec, en comparación con iOS, Android es un sistema menos vulnerable en lo que a seguridad, privacidad y vigilancia se refiere. El estudio habla de 13 vulnerabilidades graves en Android mientras que en iOS habla de 387 vulnerabilidades graves.

  Sin lugar a dudas Android es una opción perfectamente válida como sistema operativo, con algunas carencias pero con unas características excelentes y un precio más asequible que iOS.

- BlackBerry Os: El más veterano de los 4 sistemas operativos, desde su salida allá por el año 1999 cuyo único uso era mandar emails hasta ahora, sus funcionalidades han crecido y mejorado considerablemente, siendo una fantástica elección cuando se trata del ámbito empresarial. Sin embargo, para el usuario medio sigue siendo un SO con algunas carencias.

  Las aplicaciones que podemos encontrar para BlackBerry son limitadas y de baja calidad. Se trata de un sistema excelente para mandar emails y mensajería, pero en usabilidad, comparado con los demás, está un poco atrasado.

  En cuanto a rendimiento, existen algunos problemas de *lag* en según qué circunstancias. En resumen, se tratan de dipositivos muy centrados en uso empresarial, y no para el uso doméstico que pueda darle un usuario medio.

- Windows Phone: Es el más novato de los 4, y el sucesor del sistema operativo *Symbian* en los dispositivos Nokia. Windows tuvo que adaptarse, y pasar de ser un sistema operativo solo para ordenadores a ser también uno para dispositivos móviles.

  Es cierto que la adaptación fue muy lenta, no obstante, actualmente es el tercer Sistema operativo móvil más distribuido por detrás de Android e iOS.

  Se trata de un sistema operativo elegante, fácil de usar y potente, poco a poco su catálogo de aplicaciones va creciendo y con los nuevos sistemas operativos de Windows para pc, se van añadiendo nuevas funcionalidades que permiten una gran interacción entre el *smartphone* y el pc de una manera muy sencilla..

2.2.1 Mercado

En lo que a ventas se refiere, ya que Android ofrece precios más económicos y más variedad en dispositivos, es el Sistema operativo que más ventas genera, seguido por iOS, el cual tiene unos éxitos de ventas indiscutibles en Estados unidos y China.

Durante el año 2015 Android fue de manera indiscutible el sistema operativo más vendido a nivel mundial, seguido por iOS y por Windows phone.[9]

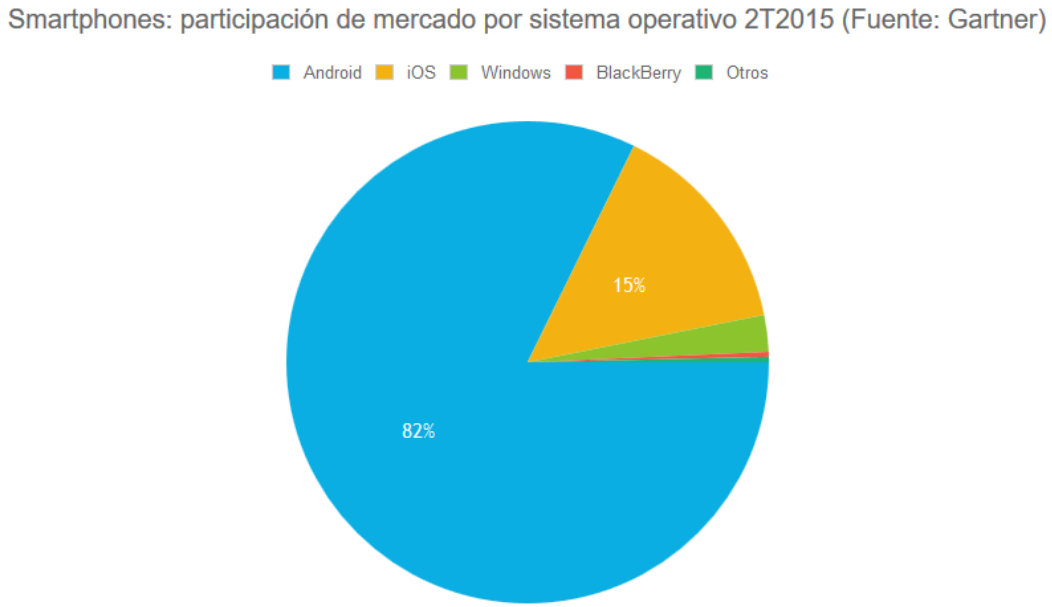


Ilustración 22. Ventas de smartphones 2015

[9] http://www.telesemana.com/blog/2015/08/25/estadisticas-mercado-de-smartphones-a-nivel-mundial/

A pesar de la diferencia en ventas a nivel de sistema operativo, a nivel de dispositivo Apple es la segunda marca más vendida, por detrás de Samsung. Esto no hace más que demostrar que Apple vende mucho, aunque no más que Android, pero debido a la poca variedad de dispositivos que posee, sus números en ventas son menores que los de Android

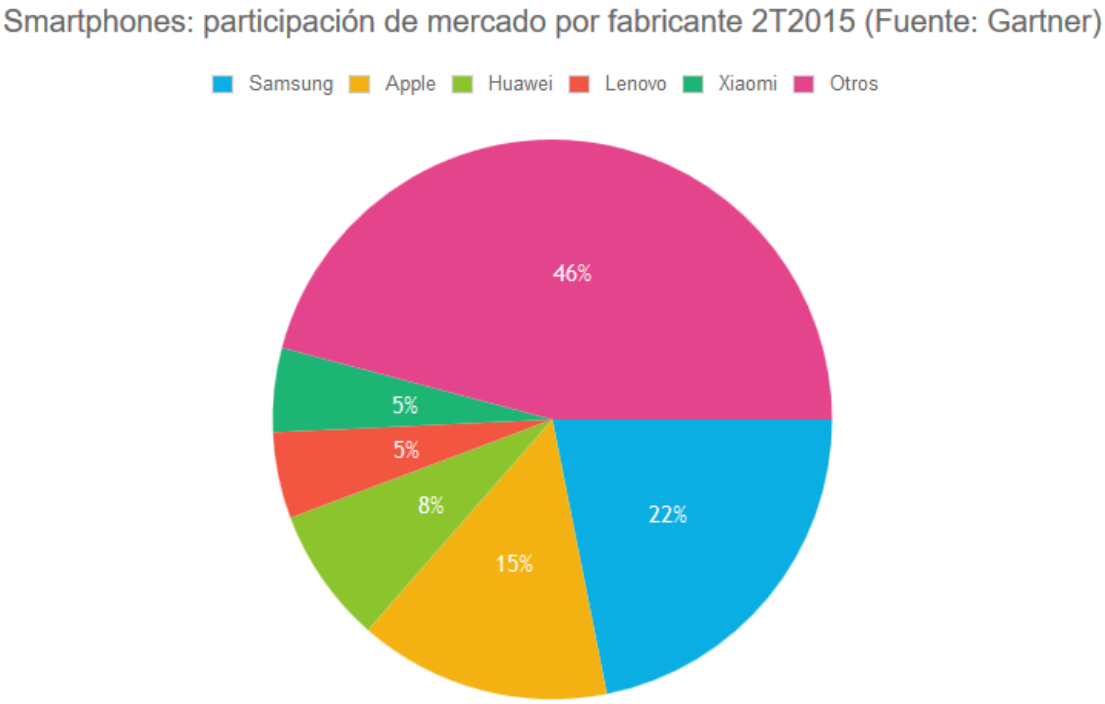


Ilustración 23. Ventas por fabricante en el año 2015

Poco a poco los *smartphone* se van afianzando en el mercado y algunos fabricantes como Nokia y su Windows Phone van afianzándose y vendiendo cada año más.

Cada año que pasa las ventas de estos dispositivos aumenta de una manera más que notable, y se prevé que en el futuro seguirán aumentando dado que cada vez tienen más éxito y son acogidos por los consumidores de manera excelente.

Ilustración 24. Ventas de Smartphones en los últimos años

2.2 Cortafuegos para Android

2.2.1 Origen del problema

Uno de las principales características de Google y de su sistema Android, es que intenta tener muchos desarrolladores facilitando, en la medida de lo posible, el desarrollo y la posterior publicación de la aplicación en su propio portal. Antes de publicar una aplicación en el *"Play store"* de Google la aplicación debe de estar firmada.

Android sólo permite la instalación de aplicaciones firmadas, pero acepta que las aplicaciones se auto-firmen con cualquier nombre, por lo que no exige una AC (Autoridad de Certificado).

Esto provoca que además de que se suban muchas aplicaciones al portal de Google, muchas de ellas pueden ser fraudulentas. Hasta que Google las detecta y las elimina de su portal pasa mucho tiempo, por lo que en la mayoría de los casos han sido descargadas ya un buen número de veces antes de su eliminación.

La mayoría de estas aplicaciones, suelen ir camufladas como si fueran juegos, cuando en realidad en segundo plano se dedican a realizar acciones de dudosa legalidad. Incluso en alguno de los casos, la aplicación incluye un temporizador para retrasar su actuación y disimular que esa aplicación es la que está provocando ese comportamiento.

Varios juegos que han sido publicados en el portal de Google fueron analizados y muchos han resultado ser troyanos camuflados como juegos, a continuación se representan los iconos de esos falsos juegos:[10]

[10] http://www.welivesecurity.com/la-es/2015/09/22/troyano-para-android-google-play/

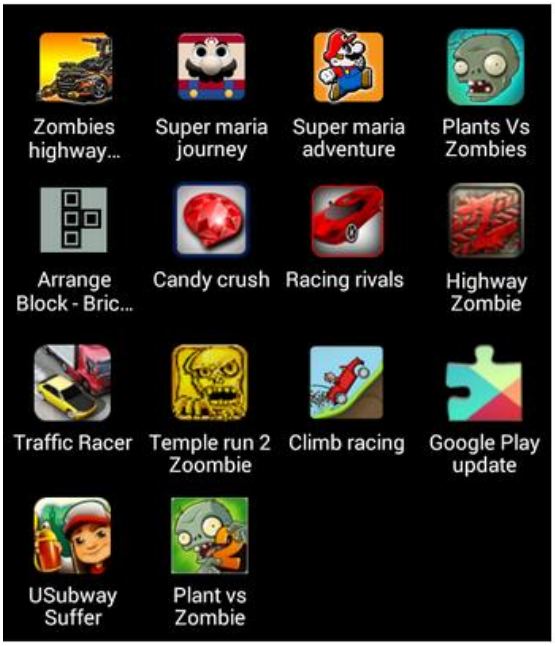


Ilustración 25. Aplicaciones infectadas

Por eso es muy importante revisar los permisos que posee una aplicación que sea de dudosa procedencia, ya que si es como en el caso de la Figura de abajo, es muy raro que una aplicación que intenta ser un juego, tenga permisos sobre los mensajes, información personal, llamadas telefónicas, etc.

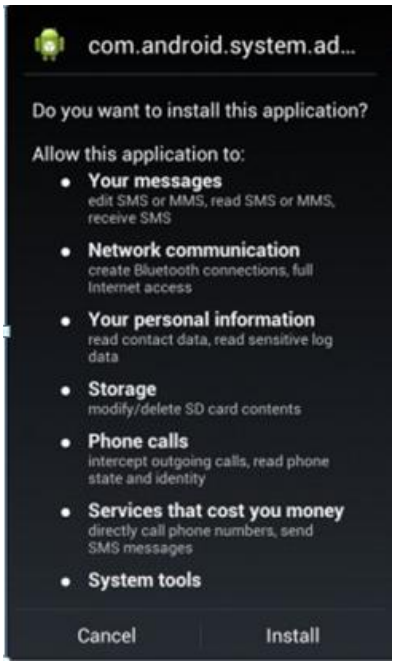


Ilustración 26. Permisos de una aplicación

Además, en Android, existen más portales de descargas de aplicaciones como *Aptoide* o *BlackMarket*. En estos portales se publican muchas aplicaciones que son de pago en el portal de Google y los publica de manera gratuita. En la mayoría de los casos, estas aplicaciones han sido fruto de loa que se denomina *"Reversing"* o

Ingeniería inversa. Es decir, la aplicación original ha sido manipulada desde su código fuente para ejecutar acciones maliciosas dentro del dispositivo donde se instala.

Con la última versión de Android (Marshmallow) se ha incluido un administrador de permisos que bloquea los permisos de las aplicaciones sobre:

- Almacenamiento
- Calendario Contactos
- Cámara
- Micrófono
- SMS
- Sensores corporales
- Teléfono
- Ubicación

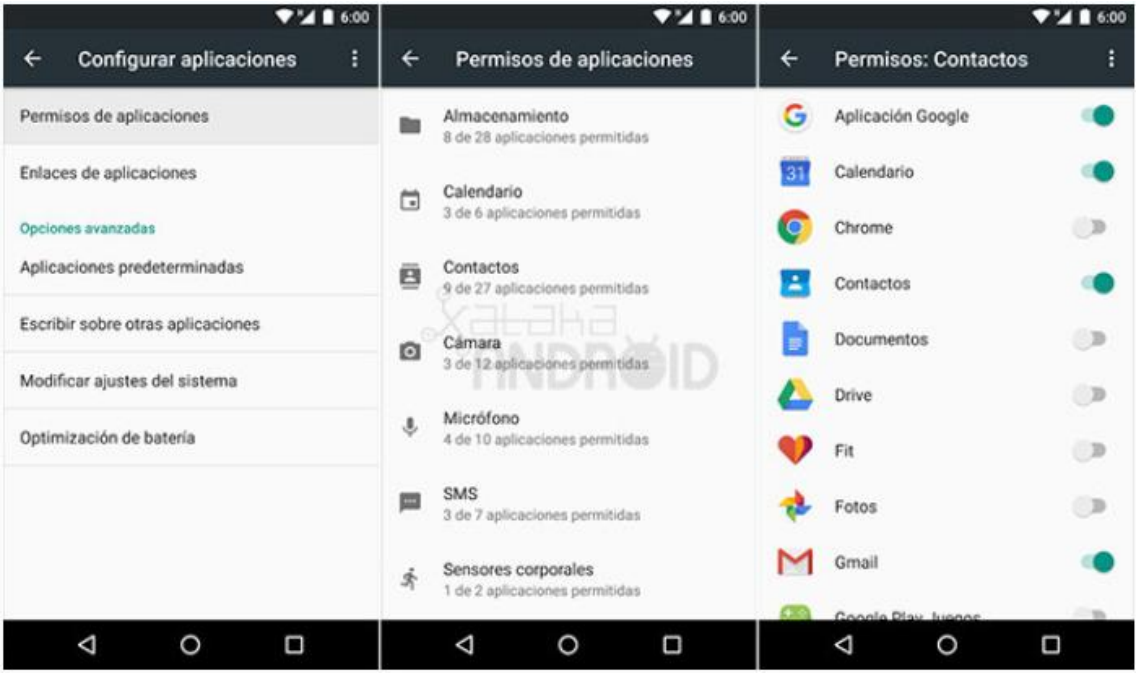


Ilustración 27. Permisos de aplicaciones en Android 6.0

Llegados a este punto, de aquí surge la idea de intentar desarrollar una aplicación que gestione el acceso entre aplicaciones. Para ello, se analiza primero el método de comunicación entre aplicaciones. Para comunicarse entre ellas las aplicaciones usan un evento denominado *"Intent"*.

2.2.2 Intent de Android

Un "*Intent*" es una operación que permite la comunicación entre aplicaciones dentro de Sistema Operativo Android. Los "*Intent*" permiten lanzar Actividades, *"Broadcast"* y servicios.[11]

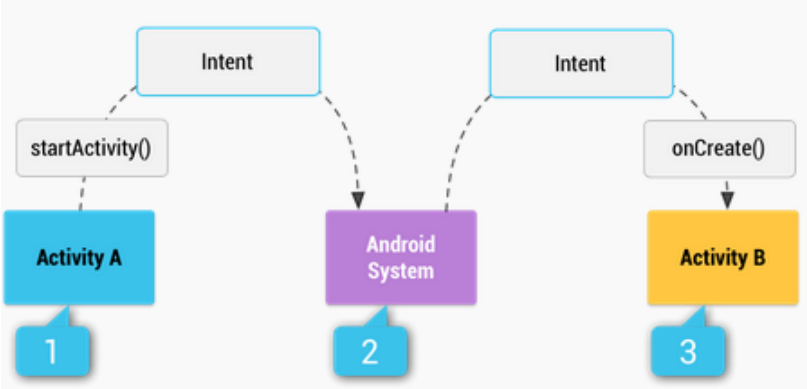


Ilustración 28. Esquema de un Intent

Un "*Intent"* está compuesto de los siguientes elementos:

- Action: La acción general que va a ser realizada.
- Data: Los datos que van a operar en la acción.
- Category: Da información adicional sobre la acción que se va a ejecutar
- Type: Especifica un tipo explícito de datos.
- Component: Especifica un tipo explícito de componente
- Extras: Proporciona información adicional.

Dentro de los "*Intent*", se pueden diferenciar en dos tipos:

- Intents Explícitos: Especifican el componente determinado al que se realiza la petición. No suelen incluir información adicional, ya que especifican que elemento desean que se ejecute. Este tipo de operación, se dan cuando por ejemplo queremos pasar una información de un formulario de una aplicación a otra aplicación.

[11] http://developer.android.com/intl/es/reference/android/content/Intent.html

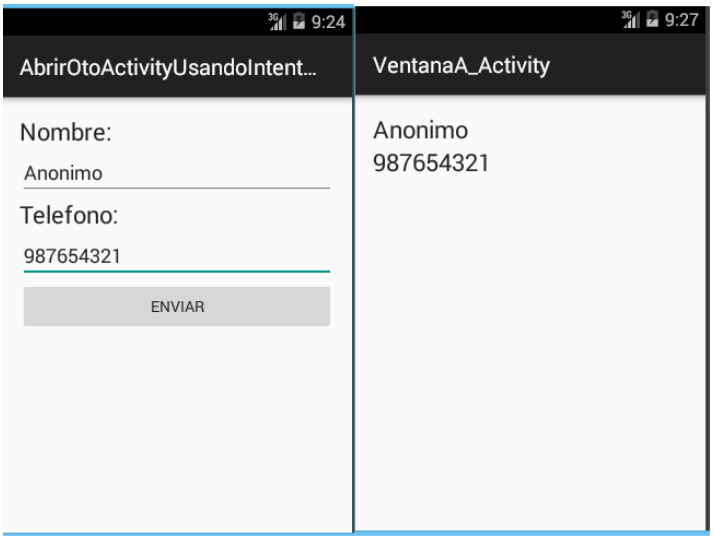


Ilustración 29. Intent explícito

- Intents Implícitos: No especifican el componente al que se le realiza la petición, pero incluyen algo más de información al sistema para determinar qué tipo de componentes pueden hacerse cargo del *"Intent"*.

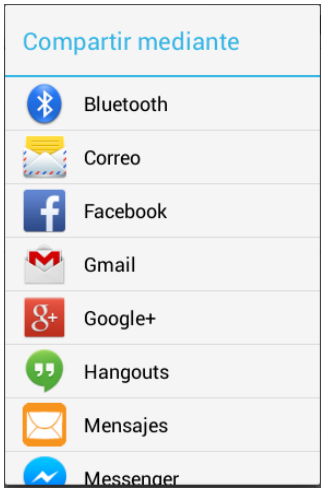


Ilustración 30. Intent Implícito

2.2.3 Intent Firewall

Tras conocer la estructura de los *"Intent"*, se buscó si existía ya alguna aplicación que cumpliera con ideas similares a las propuestas para intentar resolver el problema. Desgraciadamente no se encontró ninguna, pero si se encontró un artículo [12]que hablaba sobre un cortafuegos de *"Intent"* en Android, pero este artículo no se consideraba información oficial por parte de Google.

A partir de la versión de Android 4.4.2 (API 19) se integró dentro del sistema operativo Android un cortafuegos que gestionase los *"Intent"*. Este cortafuegos es un

[12] http://www.cis.syr.edu/~wedu/android/IntentFirewall/

componente que permite la ejecución de los *"Intent"* en función de un fichero de reglas que está escrito en formato XML.

Actualmente el cortafuegos no posee documentación oficial por parte de Google y podría cambiar con las próximas versiones del sistema operativo, por lo que ha sido muy costoso obtener información acerca de él y comprender como es su funcionamiento.

Todos los *"Intents"* surgen desde el "*framework*" de Android, incluido los que genera el propio sistema operativo. El cortafuegos posee la capacidad de denegar y de permitir los *"Intents"* y permite actualizar sus reglas de manera dinámica y en tiempo de ejecución, sin necesidad de reiniciar el dispositivo previamente.

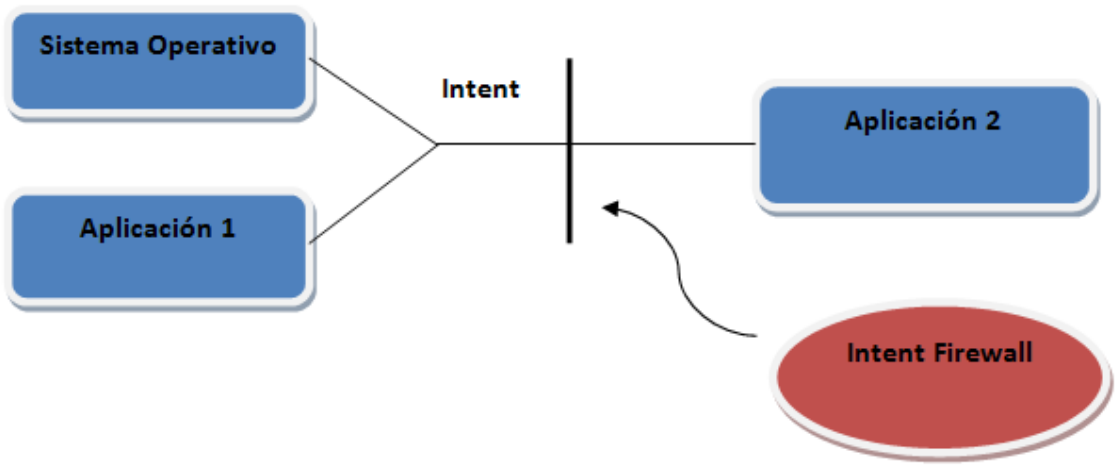


Ilustración 31. Esquema de funcionamiento del cortafuegos

La principal limitación del cortafuegos, es que sólo es accesible desde línea de comando a nivel de sistema operativo, es decir, por debajo de la capa de aplicación. Y para interactuar con el directorio donde se definen las reglas es necesario que se disponga de permisos de administrador dentro del sistema.

Incluso para los que poseen de permisos de administrador, escribir directamente en el sistema de ficheros puede suponer todo un reto, ya que la mayoría de los dispositivos Android mantienen su sistema de archivos en modo de solo lectura durante el funcionamiento normal. Esto significa que para escribir dentro del sistema de ficheros, es necesario modificar los permisos para permitir acciones de escritura, pero es necesario que una vez realizada la acción de escritura se restauren los permisos, puesto que si no se hace podría suponer una avería dentro del dispositivo.

Este conjunto de características, hacen del cortafuegos una herramienta muy frágil y compleja de utilizar, ya que puede suponer la inhabilitación del dispositivo en el caso de una mala configuración.

El cortafuegos nos permite bloquear tres tipos de *"Intents"*:

- Actividades: Bloquea el acceso a las actividades de las aplicaciones.

- Broadcast: Bloquea las señales de difusión que son emitidas por parte del sistema operativo.
- Servicios: Bloque la activación o desactivación de los servicios de las aplicaciones.

Otra de las características que nos ofrece el cortafuegos, es la posibilidad de generar logs, en ellos se puede apreciar que reglas están activas, y la información sobre los Intents que se lanzan. Para poder consultar estos logs es necesario usar la herramienta ADB (Android Debug Drive) en un ordenador mientras el dispositivo se encuentra conectado por el cable USB al ordenador.

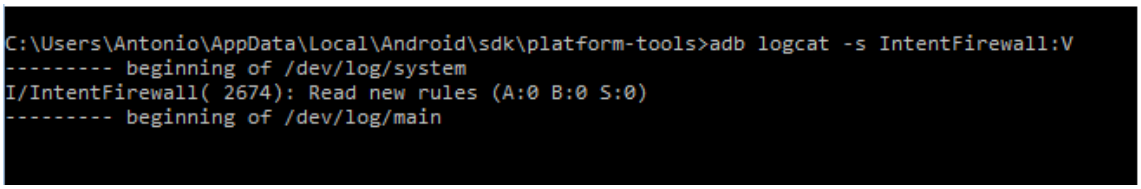


Ilustración 32. Consulta de Reglas Activas

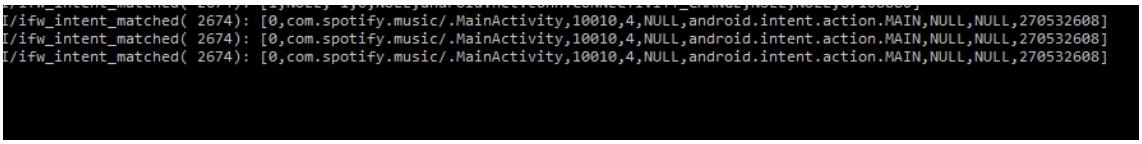


Ilustración 33. Logs de los Intents

Como se puede apreciar en la imagen de los logs, se ha intentado acceder a la aplicación *Spotify* pero el acceso ha sido denegado.

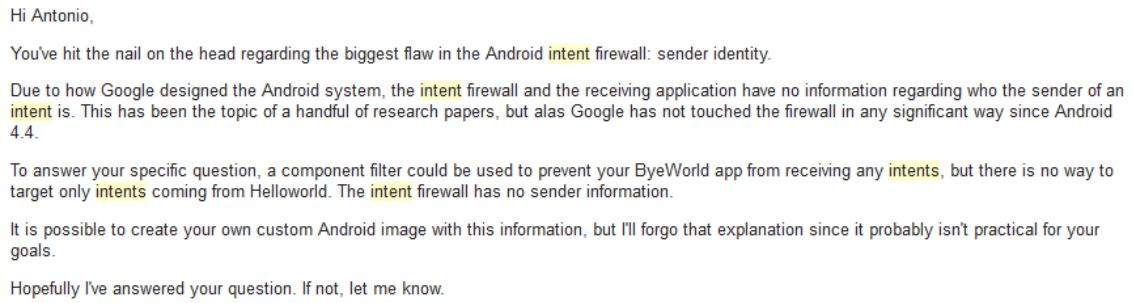
2.2.4 Problema del Intent Firewall en su versión actual

Tal y como se ha detallado unas líneas más arriba, la idea original era la de bloquear la comunicación entre aplicaciones, es decir, que por ejemplo, la aplicación X no pueda comunicarse solo con la aplicación Y, pero si con la Z.

Tras mucho probar e investigar no se consiguió ejecutar aplicaciones de manera parcelada, y dado que no existe una documentación oficial y sólo se contaba con un artículo que hablaba sobre el funcionamiento del cortafuegos, se estableció contacto vía email con el autor del artículo para determinar si efectivamente se podían bloquear los *"Intent"* en función de la aplicación que los envía.

Su respuesta, que se puede ver en la figura de abajo, determinó que era imposible bloquear un *"Intent"* en función de su emisor, ya que tal y como Google había diseñado a Android, dentro del *"Intent"* no se encuentra ningún dato que haga referencia a su emisor. Esto se ha calificado como el mayor defecto del cortafuegos.

El cortafuegos parece haber sido objetivo de muchos estudios e investigaciones, pero debido a que Google no se ha pronunciado sobre él, apenas se posee información.



Hi Antonio,

You've hit the nail on the head regarding the biggest flaw in the Android intent firewall: sender identity.

Due to how Google designed the Android system, the intent firewall and the receiving application have no information regarding who the sender of an intent is. This has been the topic of a handful of research papers, but alas Google has not touched the firewall in any significant way since Android 4.4.

To answer your specific question, a component filter could be used to prevent your ByeWorld app from receiving any intents, but there is no way to target only intents coming from Helloworld. The intent firewall has no sender information.

It is possible to create your own custom Android image with this information, but I'll forgo that explanation since it probably isn't practical for your goals.

Hopefully I've answered your question. If not, let me know.

Ilustración 34. Respuesta sobre la pregunta del cortafuegos

Finalmente, y dadas las circunstancias, se decidió por reenfocar el proyecto y tratar de construir una aplicación que interactuase con el cortafuegos y que gestionase el acceso a las aplicaciones en función de ciertas circunstancias o a través de una configuración manual preestablecida por el usuario.

De esta manera, en caso de pérdida o robo del dispositivo supondría una capa de seguridad adicional de cara al acceso de los datos almacenados en su interior. Además también sería aplicable en el caso de que si se instala una aplicación que intenta acceder a datos de otra en segundo plano sin que lo aprecie el usuario, bloqueando la aplicación objetivo, la aplicación atacante no sería capaz de obtener los datos que son accesibles desde la actividad de la aplicación.

CAPÍTULO 3

Análisis De la Aplicación

En este capítulo se especificarán los distintos requisitos con los que debe de contar la aplicación. Para definir dichos requisitos, se hará uso del estándar IEEE-STD-830-1998[13] el cual nos permite clasificarlos y describirlos mediante el uso de tablas. También se tratarán una serie de casos de uso que ilustrarán como se realiza la interacción entre el usuario y la apliación.

[13] http://www.ctr.unican.es/asignaturas/is1/IEEE830_esp.pdf

La idea de diseñar esta aplicación surge cuando se conoce la existencia de una herramienta, dentro del Sistema Operativo Android, capaz de bloquear la comunicación entre aplicaciones que se puede producir de manera voluntaria (o no) por parte del usuario que está manipulando el dispositivo, es decir, un cortafuegos de aplicaciones

El problema que la aplicación tratará de resolver es el de automatizar el comportamiento del Firewall de Android para que se pueda gestionar su uso de la manera más sencilla posible. De este modo se podrá dotar al dispositivo de un mayor grado de seguridad en el caso de perderlo, de que se instalen en él aplicaciones potencialmente maliciosas o de que el dispositivo realice un comportamiento inesperado como la activación del Wifi o del modo avión.

Para que la aplicación pueda funcionar correctamente, será necesario que el dispositivo disponga de la versión de Android 4.4.2, también conocida como *Kit Kat*, ya que sólo a partir de esa versión se encuentra el Firewall introducido dentro del Sistema Operativo. La aplicación será perfectamente válida para smartphones o para tabletas, y en general, está dirigido a todo tipo de usuarios, estén o no experimentados en el uso de aparatos electrónicos.

El lenguaje de programación usado para el desarrollo de la aplicación será Java, ya que es el único lenguaje que se usa para la elaboración de estas aplicaciones. Para el desarrollo de la interfaz y del fichero que contendrá las reglas del Firewall, se hará uso del lenguaje XML, y para la gestión y la administración de la base de datos se usará el lenguaje SQL.

3.1 Requisitos de la aplicación

Los requisitos constituyen el punto de inicio de cualquier proyecto de desarrollo de software, mediante ellos se detallará con un nivel de abstracción más bajo, la funcionalidad y las limitaciones que la aplicación.

Como ya se ha dicho anteriormente, para definir los requisitos se seguirá el estándar IEEE-STD-830-1998 que permite definir los requisitos mediante el uso de tablas.

Para definir los requisitos, los clasificaremos en 2 tipos:

- **Requisitos de Usuario**: Representan las necesidades que los usuarios expresan verbalmente.
- **Requisitos de Sistema**: Definen cual es el comportamiento del sistema en base a lo obtenido de los requisitos de usuario.

La tabla que representará cada uno de los requisitos contendrá los siguientes campos:

- **Identificador**: Con este campo se identifica el requisito, está en la parte superior de la tabla, y presenta la siguiente estructura RU-XX para los requisitos de usuario, la letra x es para el numero de requisito, en el caso de los requisitos de sistema se tiene la usa la nomenclatura RS-XX y las x corresponderán al número de requisito.
- **Nombre**: este campo es una descripción simple y breve del objetivo.
- **Descripción**: este campo detalla de forma más completa el objetivo del requisito.
- **Necesidad**: este campo indica el grado de importancia que tiene el requisito en nuestro sistema. Las necesidades se clasifican en: Alta, Media y Baja.
- **Prioridad**: este campo indica el orden de importancia de realización del requisito respecto a los otros. Las prioridades de distinguen en: Alta, Media y Baja.
- **Estabilidad**: este campo nos indica si el requisito puede variar a lo largo del desarrollo de nuestro proyecto. Distinguiremos la estabilidad en: Alta, Media y Baja.
- **Fecha**: este campo indicará la fecha de inclusión del requisito en cuestión.

Identificador	
Nombre	
Descripción	
Necesidad	
Prioridad	
Estabilidad	
Fecha	

Tabla 15. Plantilla de requisitos

3.1.1 Requisitos de Usuario

Identificador	RU - 01
Nombre	**Consistencia**
Descripción	**El funcionamiento de la aplicación debe de ser consistente, es decir, solo se bloquearán las aplicaciones que desee el usuario. Y se debe de mantener en todo momento un conocimiento sobre que aplicaciones se encuentran bloquedas.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 16. RU – 01

Identificador	RU – 02
Nombre	**Listar**
Descripción	**Deberán mostrarse todas las aplicaciones instaladas en el dispositivo**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 17. RU -02

Identificador	RU – 03
Nombre	**Configuración automática**
Descripción	**La aplicación permitirá que se modifiquen las reglas de manera automática cuando se den ciertas condiciones.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 18. RU – 03

Identificador	RU – 04
Nombre	**Configuración con wifi**
Descripción	**La aplicación podrá establecer unas reglas previamente definidas por el usuario en el caso de que se active el wifi en el dispositivo**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 19. RU – 04

Identificador	RU – 05
Nombre	**Configuración con batería**
Descripción	**La aplicación podrá establecer unas reglas previamente definidas por el usuario en el caso de que se active la carga de la betería en el dispositivo**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 20. RU - 05

Identificador	RU – 06
Nombre	**Bloqueo manual**
Descripción	**El usuario podrá bloquear, manualmente, las aplicaciones que él desee.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 21. RU – 06

Identificador	RU – 07
Nombre	**Autenticación**
Descripción	**El usuario deberá autenticarse antes de poder acceder a la aplicación**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 22. RU – 07

Identificador	RU – 08
Nombre	**Modificar contraseña**
Descripción	**El usuario podrá modificar la contraseña para acceder a la aplicación siempre que desee.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 23. RU – 08

Identificador	RU – 09
Nombre	**Eliminar reglas**
Descripción	**El usuario podrá eliminar las reglas que se encuentran activas en el momento que él desee**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 24. RU – 09

Identificador	RU– 10
Nombre	**Estructura modular**
Descripción	**La aplicación deberá de ser modular, es decir, permitirá añadir opciones nuevas a la configuración automática**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 25. RU – 10

Identificador	RU – 11
Nombre	**Sistema operativo**
Descripción	**La aplicación correrá en dispositivos con Sistema Operativo Android**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 26. RU – 11

Identificador	RU – 12
Nombre	**Protección de la contraseña de usuario**
Descripción	**La contraseña de acceso a la aplicación deberá de ser almacenada en la base de datos de manera que no pueda ser obtenida en claro.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 27. RU – 12

Identificador	RU – 13
Nombre	**Exclusión de la aplicación**
Descripción	**La aplicación que gestione el Firewall no se podrá bloquear de ninguna de las maneras.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 28. RU – 13

Identificador	RU – 14
Nombre	**Información de comportamiento automático**
Descripción	**La aplicación informará al usuario, a través de mensajes, de todo el comportamiento que se lleva a cabo por parte de la configuración automática.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 29. RU – 14

Identificador	RU – 15
Nombre	**Permisos para carpetas**
Descripción	**La aplicación deberá de ser capaz de solicitar permisos de administrador, ya que es necesario acceder a carpetas en las que originalmente no se tienen permisos**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**22/10/2015**

Tabla 30. RU – 15

Identificador	RU – 16
Nombre	**Diseño**
Descripción	**El diseño de la aplicación será sencillo y minimalista, para que nos e abuse de los recursos del dispositivo.**
Necesidad	**Alta**
Prioridad	**Media**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 31. RU – 16

3.1.2 Requisitos del sistema

Identificador	RS – 01
Nombre	**Activación de Reglas**
Descripción	**La aplicación deberá de ser capaz de establecer las reglas en el momento necesario insertando o eliminando el fichero en el directorio correspondiente.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 32. RS – 01

Identificador	RS – 02
Nombre	**Uso de listas**
Descripción	**Las aplicaciones instaladas en el dispositivo deberán mostrarse en una lista**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 33. RS – 02

Identificador	RS – 03
Nombre	**Datos en la lista**
Descripción	**Dentro de la lista cada elemento deberá mostrar el icono, el nombre y la ruta en la que se encuentra el paquete de la aplicación**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 34. RS – 03

Identificador	RS – 04
Nombre	**Base de Datos de Usuario**
Descripción	**La aplicación poseerá una base de datos para almacenar la contraseña de acceso a la aplicación definida por el usuario**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 35. RS – 04

Identificador	RS – 05
Nombre	**Base de Datos de aplicaciones**
Descripción	**La aplicación poseerá una base de datos para almacenar las aplicaciones que están bloqueadas.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 36. RS – 05

Identificador	RS – 06
Nombre	**Colores en la lista de aplicaciones a bloquear**
Descripción	**Dentro de la lista la aplicaciones que se vayan a boquear deberán marcarse en color Rojo para distinguirlas de las demás**
Necesidad	**Media**
Prioridad	**Media**
Estabilidad	**Media**
Fecha	**24/10/2015**

Tabla 37. RS – 06

Identificador	RS – 07
Nombre	**Colores en la lista de aplicaciones bloqueadas**
Descripción	**Dentro de la lista, las aplicaciones que ya se encuentran bloqueadas aparecerán marcadas en color Verde**
Necesidad	**Media**
Prioridad	**Media**
Estabilidad	**Media**
Fecha	**24/10/2015**

Tabla 38. RS – 07

Identificador	RS – 08
Nombre	**Mensajes informativos**
Descripción	**Cada vez que se realice una interacción con el firewall deberá aparecer un mensaje informativo para el usuario**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 39. RS – 08

Identificador	RS – 09
Nombre	**Establecer contraseña**
Descripción	**La aplicación deberá poseer un formulario para que se establezca la contraseña en el primer uso de la aplicación.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 40. RS – 09

Identificador	RS – 10
Nombre	**Introducir contraseña**
Descripción	**La aplicación deberá poseer un formulario para que se introduzca la contraseña cuando se quiera acceder a la aplicación.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 41. RS -10

Identificador	RS – 11
Nombre	**Modificar contraseña**
Descripción	**La aplicación deberá poseer un formulario donde se le permita al usuario establecer una contraseña nueva para acceder a la aplicación.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 42. RS – 11

Identificador	RS – 12
Nombre	**Versión de Android**
Descripción	**La aplicación soportará la versión de Android 4.4.2 y superiores.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 43. RS – 12

Identificador	RS – 13
Nombre	**Administrador**
Descripción	**El dispositivo deberá de permitir obtener permisos como Administrador dentro del sistema**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 44. RS – 13

Identificador	RS – 14
Nombre	**Almacenamiento**
Descripción	**Las reglas que se definan para la configuración automática se almacenarán en el directorio de la propia aplicación. Hasta que se requiera de su uso**
Necesidad	**Alta**
Prioridad	**Media**
Estabilidad	**Media**
Fecha	**24/10/2015**

Tabla 45. RS – 14

Identificador	RS – 15
Nombre	**Comprobación de aplicaciones**
Descripción	**La aplicación deberá de comprobar, antes de que el usuario visualice la lista de las aplicaciones, las aplicaciones que se encuentran bloqueadas en ese momento.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 46. RS – 15

Identificador	RS – 16
Nombre	**Verificación de Contraseña I**
Descripción	**Para que el usuario pueda modificar la contraseña, se deberá de comprobar previamente que el usuario conoce la contraseña que en ese momento se encuentra vigente.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 47. RS – 16

Identificador	RS – 17
Nombre	**Verificación de contraseña II**
Descripción	**El usuario deberá de introducir dos veces la contraseña nueva, para comprobar que la ha escrito correctamente.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 48. RS – 17

Identificador	RS – 18
Nombre	**Protección de la contraseña**
Descripción	**La contraseña se almacenará protegida en la base de datos de usuario haciendo uso del algoritmo SHA256.**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Media**
Fecha	**24/10/2015**

Tabla 49. RS – 18

Identificador	RS – 19
Nombre	**Permisos**
Descripción	**La aplicación deberá solicitar permisos para el uso de la batería y del wifi**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 50. RS – 19

Identificador	RS – 20
Nombre	**Mecanismos de gestión automática**
Descripción	**La aplicación constará de mecanismos que permitan gestionar de manera automática la configuración de las reglas cuando se den las situaciones necesarias**
Necesidad	**Alta**
Prioridad	**Alta**
Estabilidad	**Alta**
Fecha	**24/10/2015**

Tabla 51. RS – 20

3.1.3 Matriz de trazabilidad

Tras redactar los diferentes tipos de requisitos, se debe de comprobar que existe una continuidad o relación entre los requisitos que han sido demandados por el usuario y las características que finalmente se ha concluido que estarán en la aplicación.

Para poder realizar esta comprobación es necesaria la elaboración y el uso de una matriz de trazabilidad. En dicha matriz se puede comprobar que los requisitos de usuario están relacionados con al menos un requisito del sistema.

La Matriz de Trazabilidad de Requisitos ayuda a realizar un seguimiento a los requisitos a lo largo del ciclo de vida del proyecto para asegurar que se están cumpliendo de manera eficaz.

La matriz que se muestra en la Tabla 52, demuestra lo establecido previamente en el párrafo anterior obteniendo un resultado satisfactorio. No obstante esto no implica que no se puedan modificar o añadir nuevos requisitos tanto de usuario como del sistema en un futuro.

	RU - 01	RU - 02	RU - 03	RU - 04	RU - 05	RU - 06	RU - 07	RU - 08	RU - 09	RU - 10	RU - 11	RU - 12	RU - 13	RU - 14	RU - 15	RU - 16
RS - 01			X	X	X	X			X							
RS – 02		X		X	X											X
RS – 03		X											X			
RS – 04							X	X								
RS – 05	X			X	X	X			X				X			
RS – 06	X	X														X
RS – 07	X	X														X
RS – 08			X	X	X	X			X					X		
RS – 09																X
RS – 10							X									X
RS – 11								X								X
RS – 12											X					
RS – 13			X			X									X	
RS – 14			X	X	X											

RS – 15	X	X														
RS – 16		X						X								
RS – 17								X								
RS – 18							X	X				X				
RS – 19				X	X											
RS - 20			X	X	X					X						

Tabla 52. Matriz de trazabilidad

Con el resultado final de la matriz, podemos comprobar que todos los requisitos del sistema provienen de uno o de varios requisitos de usuario, por lo que todos están relacionados.

3.2 Casos de Uso

En este apartado se muestran los casos de uso de la aplicación. Un caso de uso es una secuencia de interacciones que se desarrollan entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso especifican la comunicación y el comportamiento de un sistema a través de su interacción con los usuarios o con otros sistemas.

Generalmente, en los casos de uso se incluye el tipo de actor, que es el que protagoniza el caso de uso. Para los casos de uso descritos a continuación, ese elemento se categorizará en dos posibles tipos, usuario autenticado o usuario nuevo

Para la representación de los distintos casos de uso, se usará una tabla estándar que se definirá a continuación, además de unas ilustraciones que ayudarán a su entendimiento.

Las tablas poseerán los siguientes campos:

- **Identificador**: Identificará de manera única a cada uno de los casos de uso. La nomenclatura que se seguirá será CU – XX siendo XX un número que empezará en el valor 01 e irá incrementándose.
- **Título**: Nombre del caso de uso
- **Descripción**: Detallará el caso de uso a comentar
- **Precondiciones**: Condiciones previas que se deben de dar antes de iniciar el caso de uso
- **Postcondiciones**: Condiciones que se producirán tras realizar el caso de uso.
- **Escenario principal**: Se describe la secuencia de interacciones de manera ordenada entre el usuario y el sistema
- **Escenario alternativo:** Describirá las condiciones del Caso de uso en caso de error.

Identificador	CU - XX
Título	
Descripción	
Precondiciones	
Postcondiciones	
Escenario principal	
Escenario alternativo	

Tabla 53. Plantilla de Casos de Uso

Identificador	CU – 01
Título	Establecer contraseña.
Descripción	El usuario accede por primera vez a la aplicación y establece una contraseña para acceder a ella.
Precondiciones	Iniciar la aplicación por primera vez.
Postcondiciones	Se accederá al menú de la aplicación.
Escenario principal	1. El usuario abre la aplicación. 2. El usuario introduce la contraseña que desea establecer en el campo de texto. 3. El usuario pulsa sobre el botón con texto "Establecer contraseña".
Escenario alternativo	1. El usuario no introduce ninguna contraseña 2. El usuario pulsa el botón 3. La aplicación muestra un mensaje indicando que no se ha introducido una contraseña y que esta debe de ser introducida.

Tabla 54. CU – 01

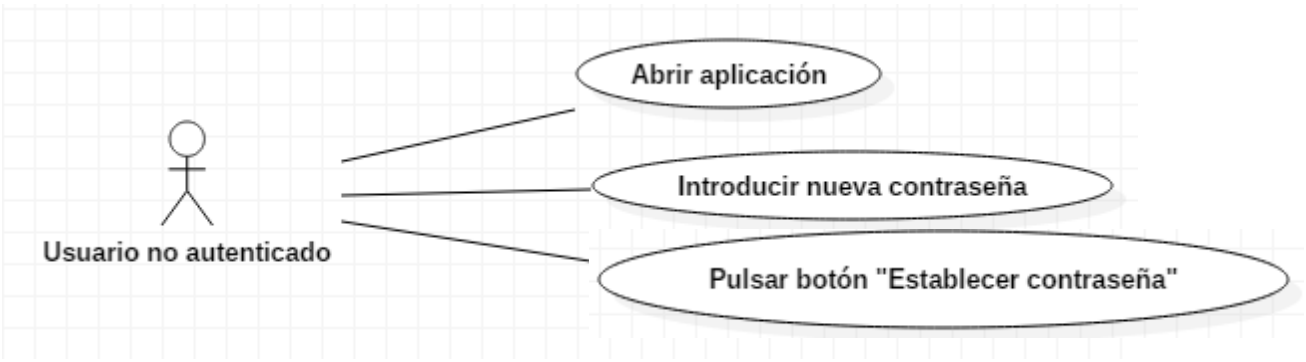


Ilustración 35. CU – 01

Identificador	CU – 02
Título	Introducir contraseña.
Descripción	El usuario accede a la aplicación e introduce la contraseña para dirigirse al menú de la aplicación.
Precondiciones	Iniciar la aplicación después del primer uso.
Postcondiciones	Se accederá al menú de la aplicación.
Escenario principal	1. El usuario abre la aplicación. 2. El usuario introduce la contraseña que desea establecer en el campo de texto. 3. El usuario pulsa sobre el botón con texto "Acceder".
Escenario alternativo	1. La contraseña introducida no es la correcta. 2. La aplicación muestra un mensaje indicando que la contraseña introducida no es la correcta.

Tabla 55. CU – 02

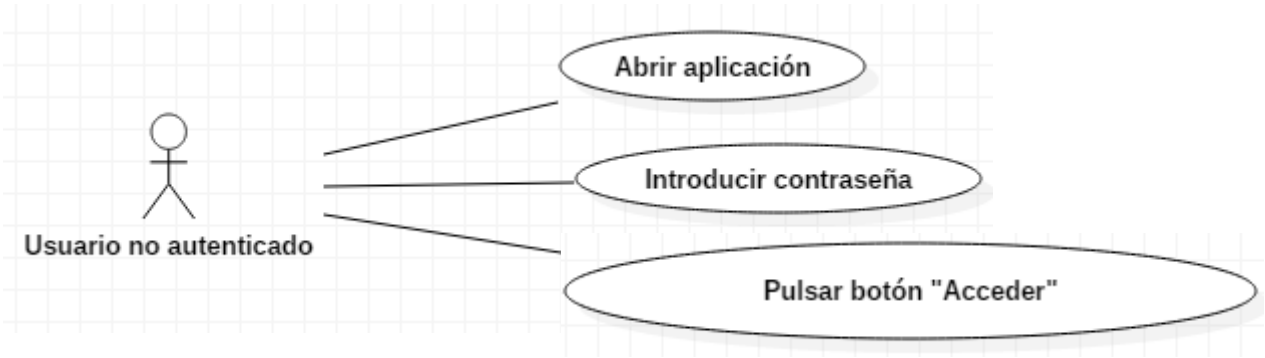


Ilustración 36. CU - 02

Identificador	CU – 03
Título	Configuración manual.
Descripción	El usuario establece cuales de las aplicaciones que se encuentran instaladas en el dispositivo se bloquean en ese mismo instante.
Precondiciones	Haber accedido al menú de la aplicación.
Postcondiciones	La aplicación establece las reglas, bloquea las aplicaciones seleccionadas y muestra al usuario un mensaje indicándole que la tarea se ha realizado correctamente.
Escenario principal	1. El usuario, que se encuentra en el menú, pulsa sobre el botón “Configuración manual”. 2. El usuario se desplaza sobre la lista mostrada y selecciona las aplicaciones que desea bloquear. 3. El usuario pulsa sobre el botón “Bloquear apps seleccionadas”. 4. El usuario acepta conceder a la aplicación permisos de administrador.
Escenario alternativo	1. No se conceden permisos de administrador 2. La aplicación muestra un mensaje indicando que no se ha realizado correctamente la operación.

Tabla 56. CU – 03

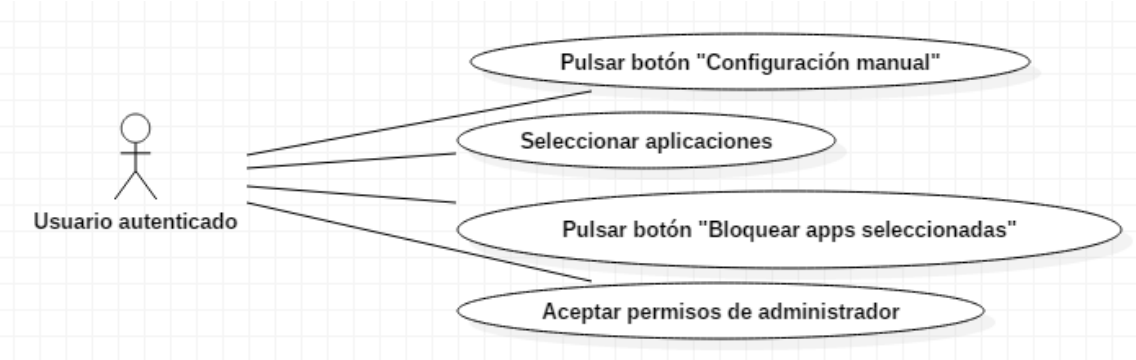


Ilustración 37. CU - 04

Identificador	CU – 04
Título	Configuración para Batería/Wifi.
Descripción	El usuario establece que aplicaciones se bloquearán cuando el dispositivo empiece a cargar la batería o cuando se active la conectividad a internet a través del Wifi.
Precondiciones	Haber accedido al menú de la aplicación.
Postcondiciones	La aplicación crea el fichero con las reglas y lo almacena en el directorio de la aplicación, adicionalmente muestra un mensaje indicando que las reglas se han almacenado correctamente.
Escenario principal	1. El usuario, que se encuentra en el menú de la aplicación, pulsa sobre el botón “Configuración con Wifi” o “configuración con Batería”. 2. El usuario se desplaza sobre la lista mostrada y selecciona las aplicaciones que desee bloquear. 3. El usuario pulsa sobre el botón “Bloquear apps seleccionadas”.
Escenario alternativo	

Tabla 57. CU – 04

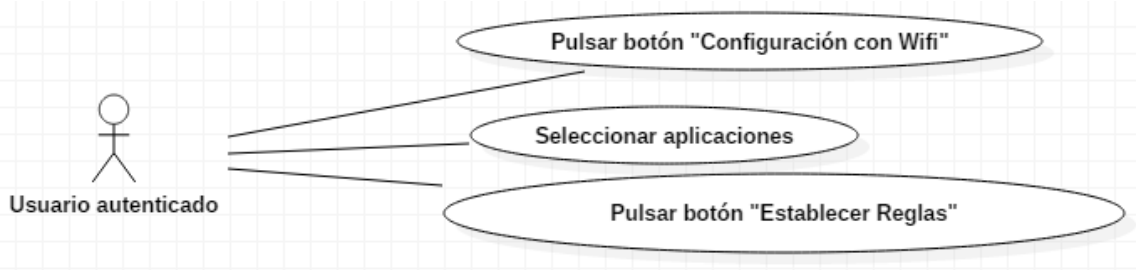


Ilustración 38. CU – 05

Identificador	CU - 05
Título	Modificar contraseña.
Descripción	El usuario decide modificar la contraseña que gestiona el acceso a la aplicación.
Precondiciones	Haber accedido al menú de la aplicación.
Postcondiciones	La aplicación modificará la contraseña almacenada en la base de datos y avisará con un mensaje al usuario de que el cambio se ha producido correctamente.
Escenario principal	1. El usuario, que se encuentra en el menú pulsa sobre el botón de “Cambiar contraseña”. 2. El usuario introduce la contraseña actual. 3. El usuario introduce la contraseña nueva. 4. El usuario introduce nuevamente la contraseña nueva. 5. El usuario pulsa sobre el botón “Cambiar contraseña”.
Escenario alternativo	1. El usuario introduce una contraseña errónea. 2. La aplicación mostrará un mensaje de error indicando que la contraseña introducida es errónea.
Escenario alternativo	1. Las dos contraseñas nuevas introducidas por el usuario no coinciden. 2. La aplicación mostrará un mensaje de error indicando que la contraseña introducida es errónea.

Tabla 58. CU – 05

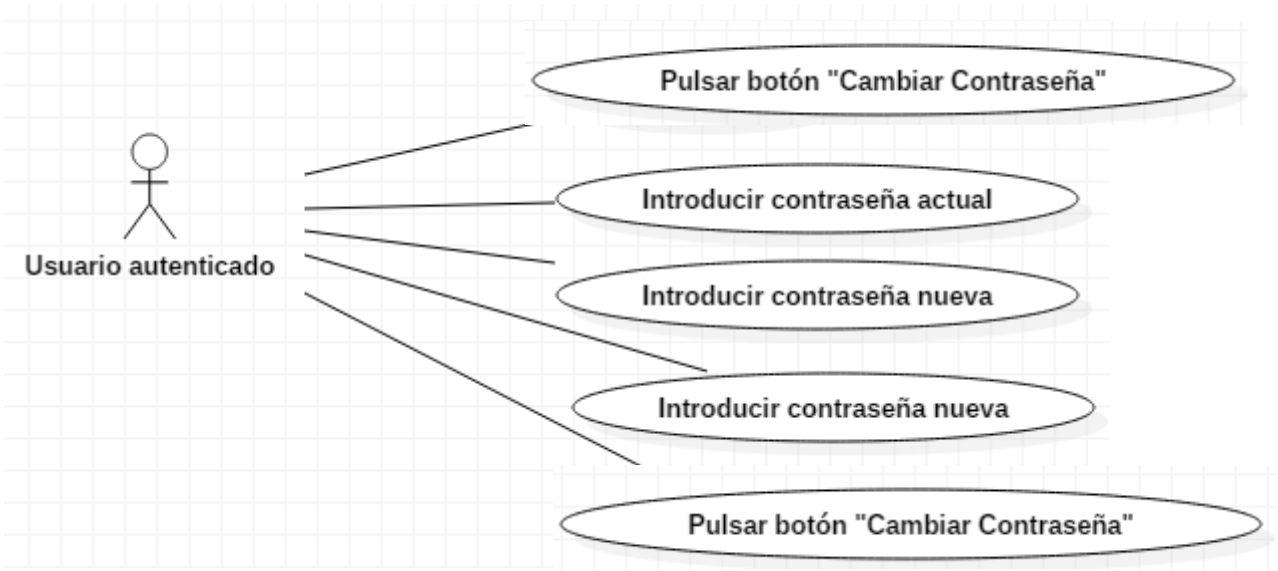


Ilustración 39. CU – 05

Identificador	CU – 06
Título	Activar/Desactivar configuración automática
Descripción	El usuario decide activar/desactivar la opción de configuración automática.
Precondiciones	Haber accedido al menú de la aplicación.
Postcondiciones	La aplicación activa/desactiva los mecanismos que gestionan el comportamiento automático de la aplicación y muestra al usuario un mensaje de que la acción se ha realizado correctamente
Escenario principal	1. El usuario, que se encuentra en el menú pulsa sobre el botón de "Activar/Desactivar configuración automática"
Escenario alternativo	

Tabla 59. CU – 06

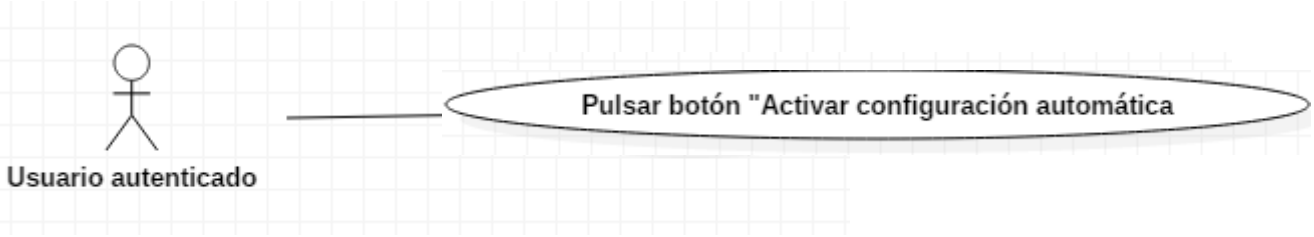


Ilustración 40. CU – 06

Identificador	CU – 07
Título	Limpiar Reglas
Descripción	El usuario decide eliminar todas las reglas que se encuentran activas en el dispositivo
Precondiciones	Haber accedido al menú de la aplicación.
Postcondiciones	La aplicación elimina las reglas y muestra un mensaje indicándole al usuario de que las reglas se han eliminado correctamente
Escenario principal	1. El usuario, que se encuentra en el menú pulsa sobre el botón de "Eliminar Reglas"
Escenario alternativo	

Tabla 60. CU – 07

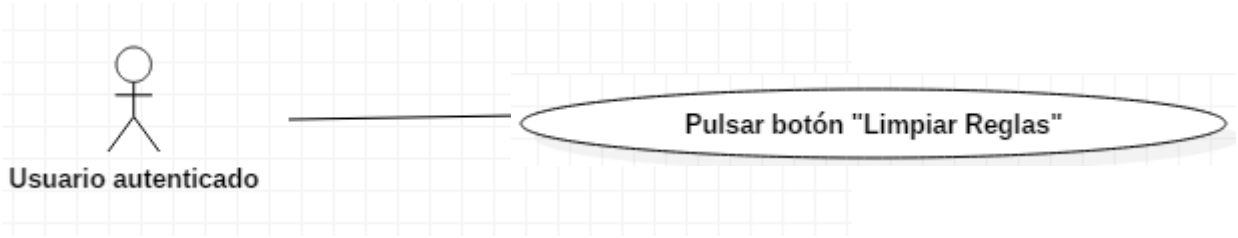


Ilustración 41. CU - 07

CAPÍTULO 4

Diseño De la Aplicación

Tras la toma de requisitos y la definición de casos de uso, en este capítulo se abordarán todos los aspectos referentes al diseño de la aplicación. Para poder llevarlo a cabo de una manera eficiente se incluirán diagramas de alto nivel que ayuden a la comprensión acerca de cómo se ha implementado y cómo se realizan las principales tareas que realiza la aplicación.

4.1 Arquitectura del Sistema

Como ya se había acordado previamente, la aplicación deberá de gestionar el cortafuegos de aplicaciones que viene implementado en las versiones del sistema operativo Android a partir de la versión 4.4.2.

Esta aplicación, en base a lo acordado previamente, deberá de poseer una interfaz que permita al usuario interactuar de la manera más sencilla posible con el cortafuegos, dando la posibilidad de bloquear las aplicaciones que el usuario desee de manera cómoda, siendo la aplicación la que se encargue de la gestión de las reglas a establecer.

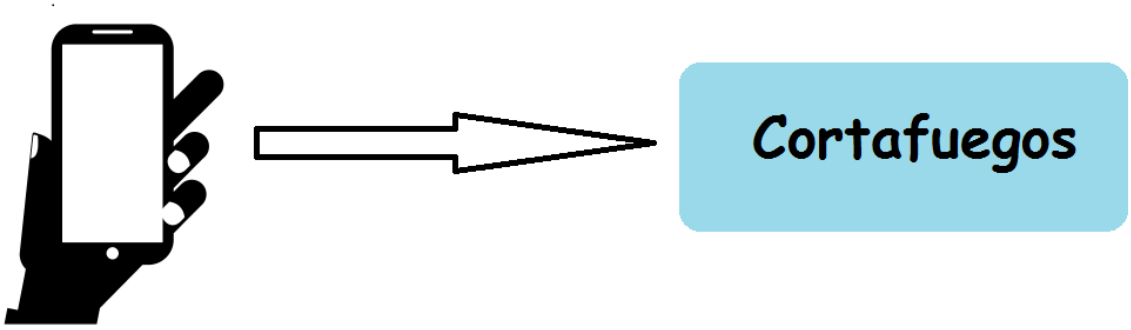


Ilustración 42. Arquitectura del sistema

Tras analizar cómo debería de funcionar y las características que debería de mostrar la aplicación, se opta por usar el patrón de arquitectura Modelo-Vista-Controlador.

Este patrón separa los datos y la lógica de negocio de la interfaz de la aplicación, de esta manera nos permite dividir más fácilmente el trabajo y nos ayuda a realizar una aplicación que pueda desarrollar futuros componentes o librerías que puedan ser reutilizadas en un futuro.

A continuación se detalla cada uno de los componentes del patrón Modelo-Vista-Controlador:

- **Modelo:** Cuando hablamos de modelo, hacemos referencia a las representaciones que son elaboradas en base a la información con la que trabaja la aplicación. Esto sería algo análogo a los beans y nos permite elaborar código reutilizable. Dentro del marco del modelo, se engloba también el tipo de herramienta que se usará para almacenar la información que usará la aplicación (Bases de datos, web services, etc).
- **Vista:** La vista responde a la parte de la aplicación que interactuará de manera directa con el usuario, es decir, la interfaz.
- **Controlador:** Este componente acoge a todas las clases que nos ayudarán a gestionar la lógica de la aplicación, de la respuesta a eventos, peticiones de usuario, etc. Este componente también puede mandar comandos a la

vista asociada si se considera necesario un cambio en la forma en la que se representa el modelo.

Para terminar de comprender como funciona este patrón de arquitectura, se puede consultar la ilustración 43 que se encuentra a continuación.

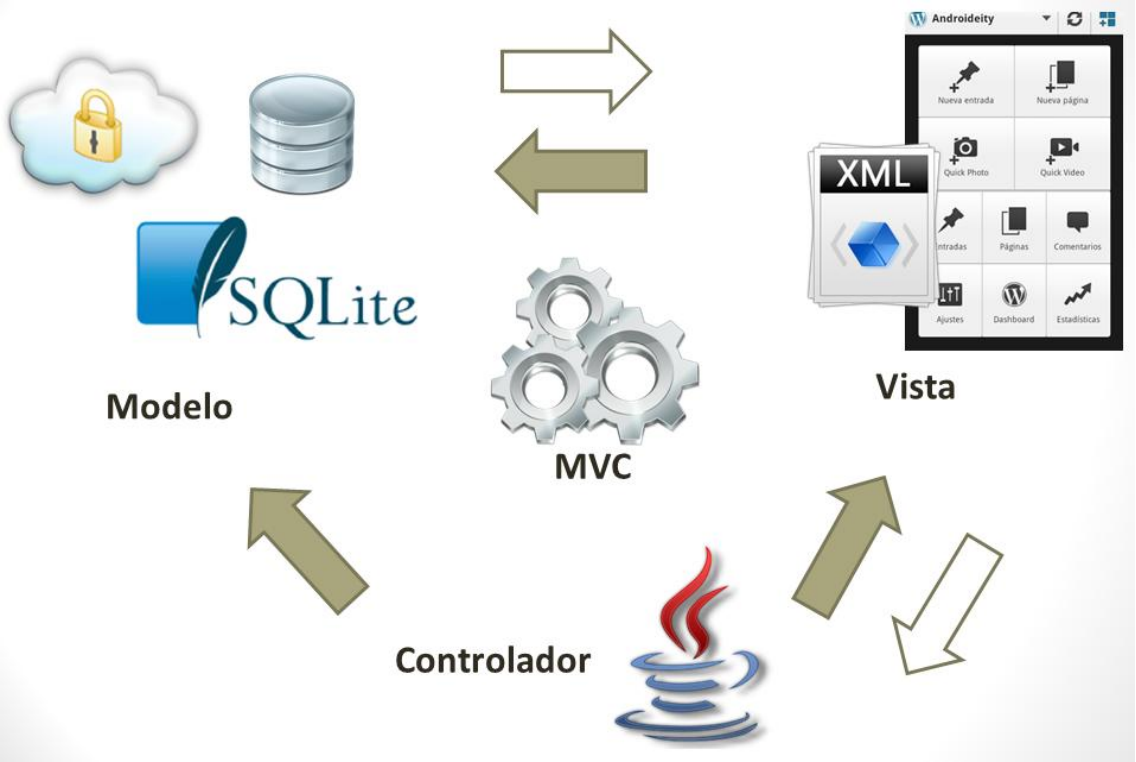


Ilustración 43. Esquema del patrón Modelo-Vista- Controlador

A continuación se va a proceder a detallar cada uno de los elementos explicados anteriormente, para facilitar su comprensión se detallarán de manera separada, y se comentará donde se produce la interacción entre cada uno de los elementos que componen la arquitectura Modelo-Vista-Controlador.

4.1.1 Modelo

Para poder describir correctamente este elemento, se hace uso del modelo Entidad/Relación, donde podemos ver la estructura de la base de datos en la que se almacena. Para este caso, el modelo está compuesto de una base de datos compuesta de dos tablas. En una tabla se almacenan las aplicaciones que se encuentran bloqueadas o anotadas en algunas de las configuraciones que ofrece la aplicación, y en la otra tabla se aloja la clave de acceso a la aplicación que se almacena protegida a través de un algoritmo de una función resumen.

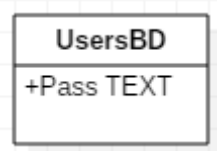


Ilustración 44. Estructura BBDD usuarios

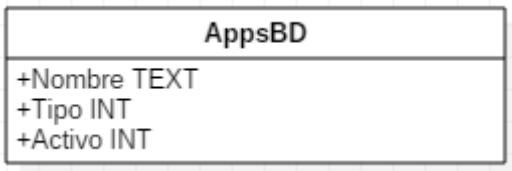


Ilustración 45. Estructura BBDD aplicaciones

Dado que ambas tablas almacenan datos que son totalmente independientes, no es necesario que guarden algún tipo de relación entre ellas.

La tabla de usuarios sólo dispone de un campo en el que se almacenará la contraseña de acceso a la aplicación. Por el contrario, la Base de datos de aplicaciones contendrá los siguientes campos:

- Nombre: Nombre de la aplicación
- Tipo: Tipo de regla a la que pertenece la aplicación (bloqueo manual, con el uso del wifi o con la carga de la batería).
- Activo: Campo que nos indica si la aplicación se encuentra en ese momento bloqueada o no.

4.1.2 Vista

Para detallar este elemento con claridad, se va a usar el diagrama de clases que se encuentra en la ilustración 46. A continuación se realizará una descripción de cada una de las clases que intervienen en este punto.

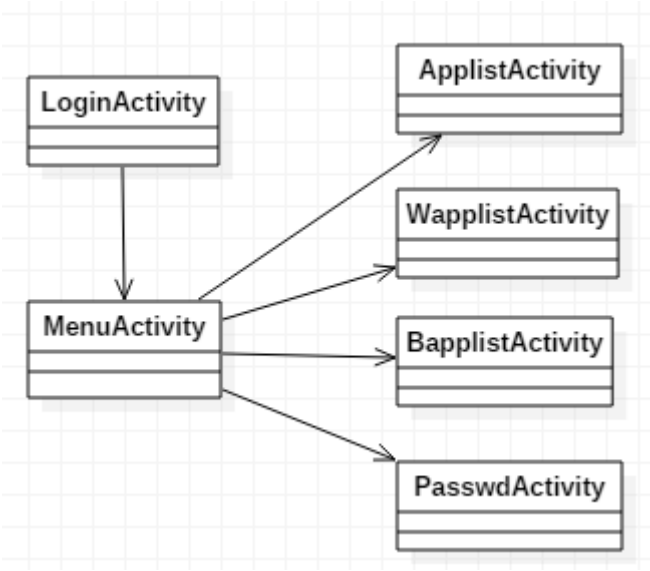


Ilustración 46. Diagrama de relación de las interfaces

- **LoginActivity:** Esta actividad será la que gestione el acceso a la aplicación a través de un formulario donde se le solicitará al usuario que introduzca la contraseña de la aplicación.

- **MenuActivity:** En esta actividad se le da a elegir al usuario entre las distintas funcionalidades que ofrece la aplicación.

- **AppListActivity:** Esta interfaz muestra al usuario un listado con todas las aplicaciones instaladas en el dispositivo para que él seleccione y bloquee de manera manual las aplicaciones que desee

- **WaplistActivity:** Esta interfaz es muy similar a la anterior, con la diferencia de que en este caso las aplicaciones que se deciden bloquear no se bloquean en ese mismo momento, si no cuando se active el Wifi en el dispositivo.

- **BapplistActivity:** Igual que la interfaz anterior, solo que esta vez las reglas se establecerán en el momento en el que se encuentre cargándose la batería

- **PasswdActivity:** Esta actividad nos mostrará un formulario que permitirá al usuario modificar la contraseña de acceso a la aplicación

4.1.3 Controlador

Finalmente, el último componente a detallar es el controlador. En este componente se encuentran las clases encargadas de realizar las mayores operaciones de cómputo. A continuación se ofrece una descripción de las clases que componen el controlador, y en la Ilustración 47 se puede visualizar el diagrama de clases.

- **Login**: Esta clase es la que gestiona la validación de la contraseña o su inserción en la base de datos en el caso de que sea el primer uso de la aplicación. Hereda de la clase APPCompatActivity.

- **Menu**: En esta clase en java se gestiona el acceso a las diferentes actividades que ofrece la aplicación. También se realizan otras actividades como activar o desactivar la configuración automática o limpiar las reglas que se encuentran activas en ese momento en el cortafuegos.

- **Applist**: Es la clase encargada de mostrar al usuario la lista de aplicaciones para posteriormente, bloquearlas de manera manual. Esta clase hereda de la clase ListActivity.

- **Wapplist**: Esta clase es muy similar a la anterior, pero en vez de establecer las reglas, las almacena en el dispositivo para que se establezcan en el momento en el que se activa el Wifi en el dispositivo. Esta clase hereda de la clase ListActivity.

- **Bapplist:** Exactamente igual que la clase anterior, con la diferencia de que esta clase genera unas reglas para cuando se activa la carga de la batería en el dispositivo.

- **AplicationAdapter:** Clase auxiliar que ayuda a la generación de las listas en las clases detalladas anteriormente. Hereda de la clase Array adapter.

- **ReadManifest**: Clase que contiene un conjunto de métodos que permiten parsear código XML de los ficheros.

- **Apps:** Clase usada para construir el objeto app.

- **PasswdActivity:** Esta clase gestiona las tareas necesarias para realizar la modificación de la contraseña que da acceso a la aplicación, así como algunas comprobaciones previas, como puede ser la comprobación de que se conoce la contraseña actual para poder modificarla.

- **MyReceiver:** Esta clase se usa para la gestión de la configuración automática. Gracias a ella se podrá conocer cuando se dan ciertas circunstancias que la aplicación deberá de identificar para establecer las reglas previamente establecidas. Esta clase hereda de otra clase denominada “BroadcastReceiver.

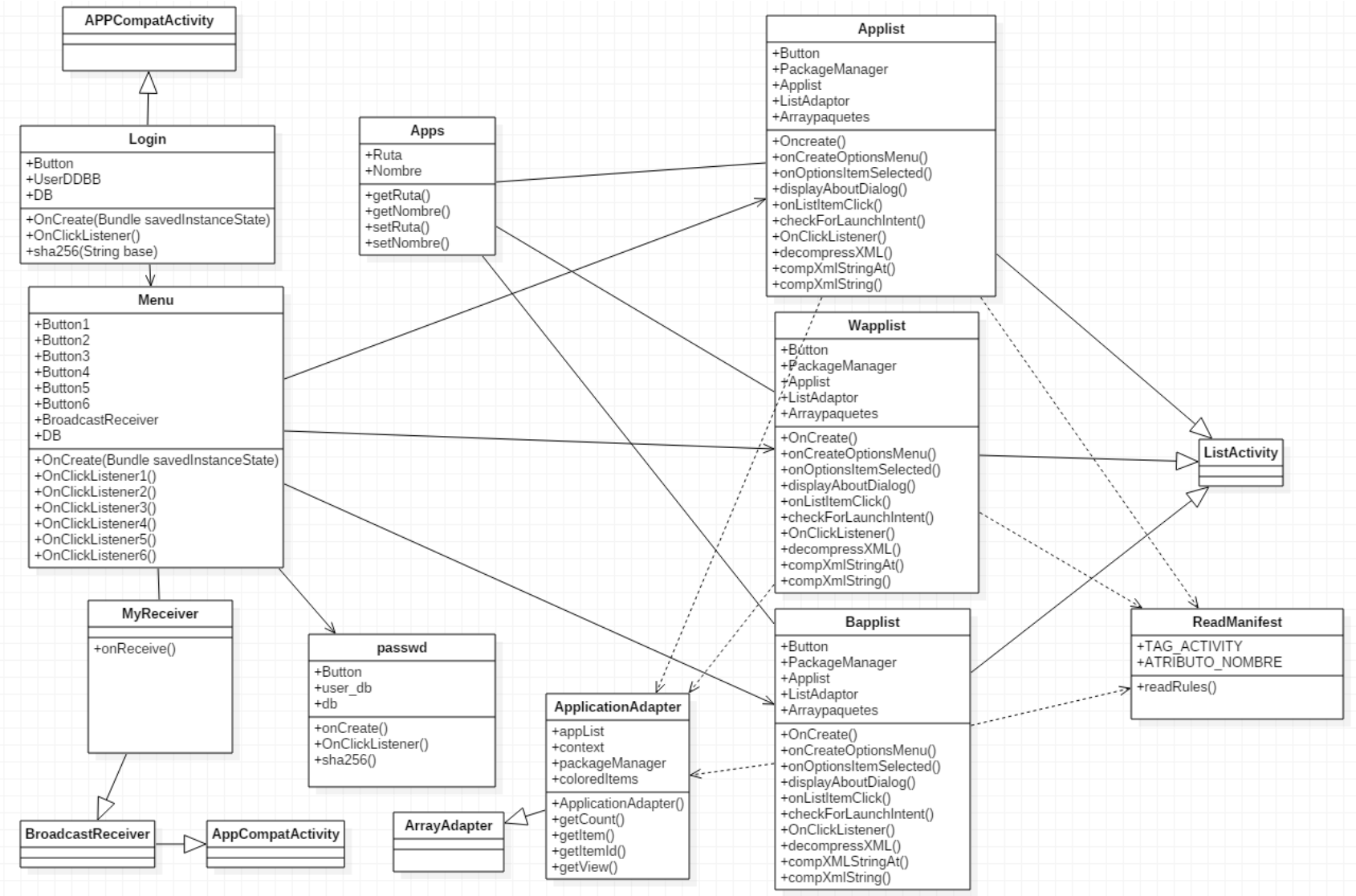


Ilustración 47. Diagrama de clases

4.2 Diagramas de Secuencia

Tras analizar el diagrama de clases y comprobar las relaciones que hay entre ellas, toca hablar de los diagramas de secuencia. Estos diagramas representan la secuencia de operaciones que son necesarias para levar a cabo una tarea. Gracias a los diagramas de secuencia se puede apreciar con claridad la relación entre clases y la interactuación mediante los métodos y los mensajes.

Para cada funcionalidad se ha escogido una tarea representativa que será llevada a cabo por parte del usuario que hace uso del sistema.

A continuación, se mostrará un diagrama de secuencia por cada uno de los casos de uso que se describieron durante el apartado de análisis de la aplicación:

- CU - 01: Establecer contraseña
- CU – 02: Introducir contraseña
- CU – 03: Configuración manual
- CU – 04: Configuración para Batería/wifi
- CU – 05: Modificar contraseña
- CU – 06: Activar/Desactivar configuración automática
- CU – 07: Limpiar reglas

En la Ilustración 48 se puede apreciar el primer diagrama, en él el usuario establece por primera vez la contraseña de acceso a la aplicación.

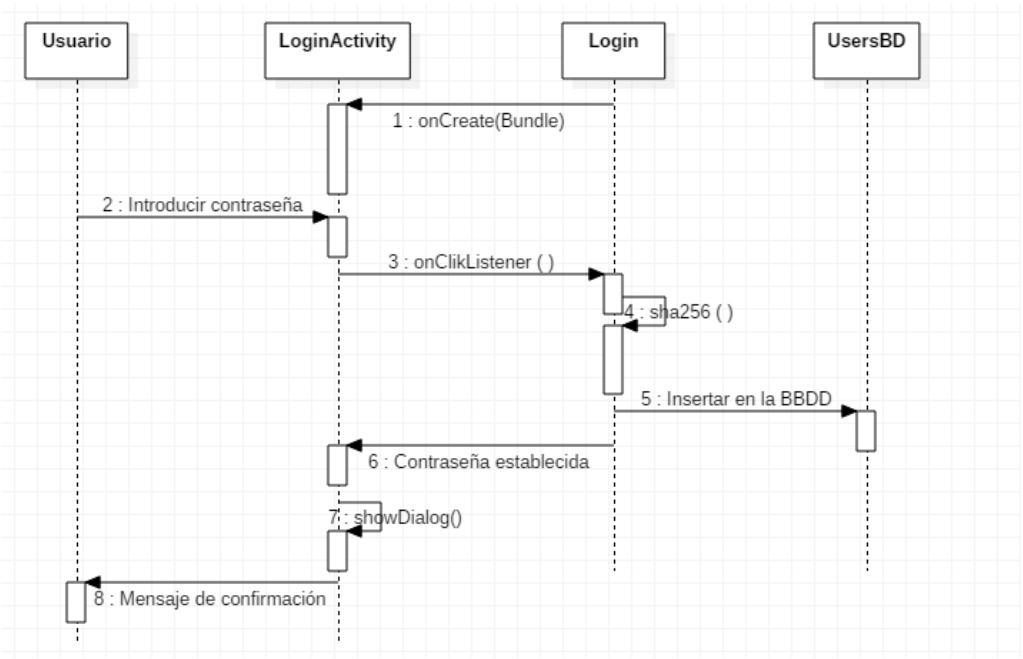


Ilustración 48.Diagrama de secuencia establecer contraseña

En la Ilustración 49, se puede ver el diagrama de secuencia que refleja cómo sería la interacción entre los distintos elementos cuando se da el caso de logarse en el sistema para acceder a la aplicación.

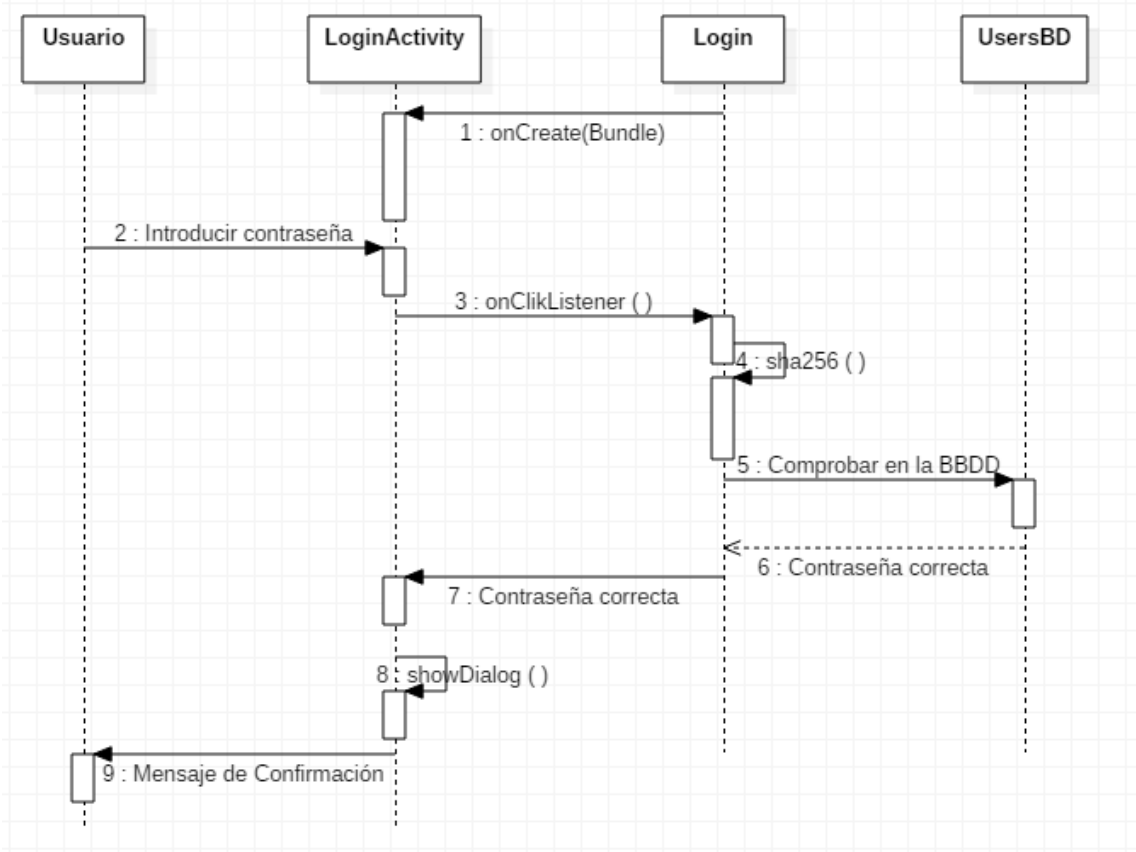


Ilustración 49. Diagrama de secuencia Introducir contraseña

El siguiente diagrama de secuencia, muestra las interacciones que realizan entre sí los componentes del sistema cuando se quiere realizar el bloqueo manual de las aplicaciones que tenemos en el dispositivo

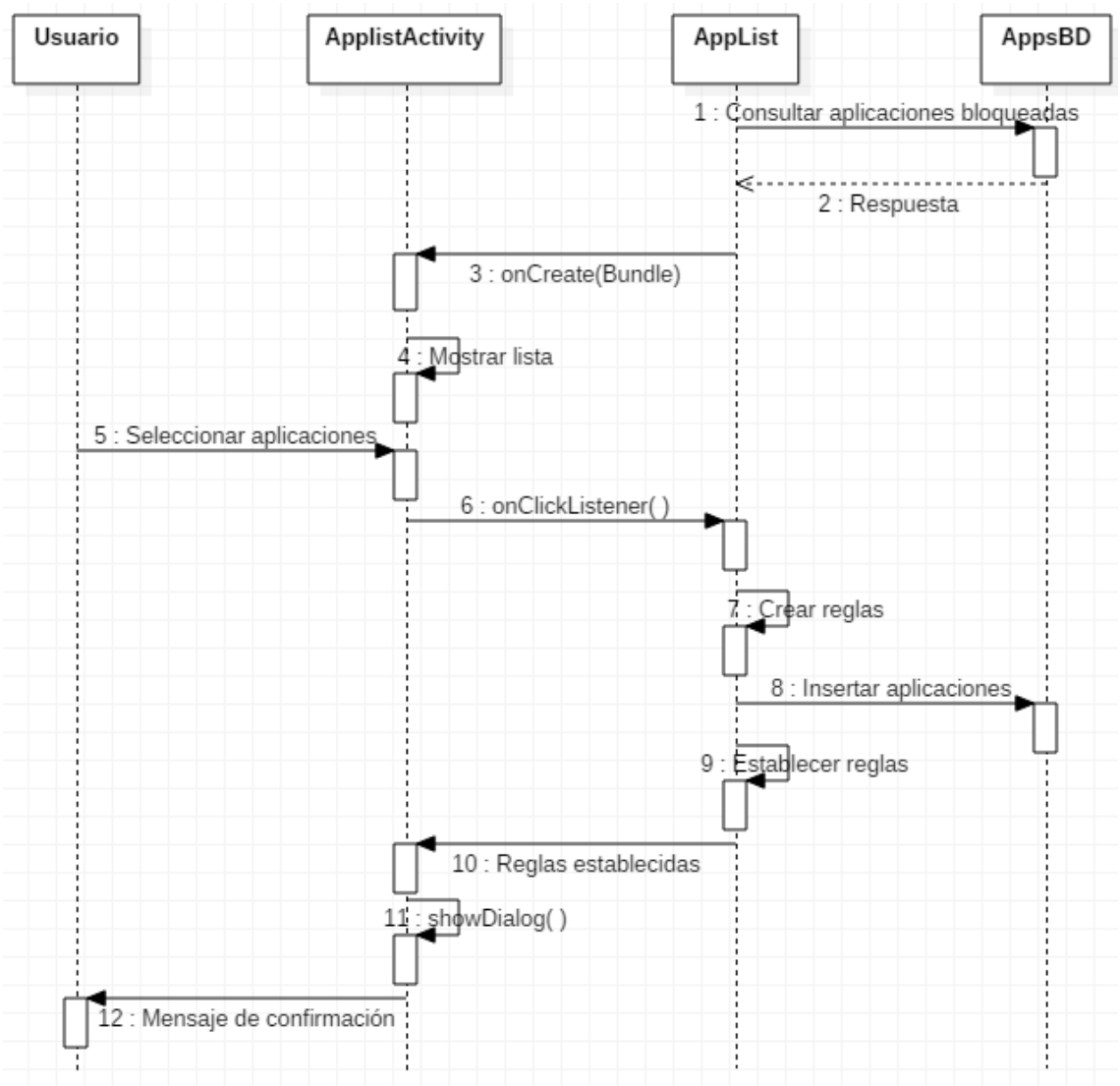


Ilustración 50. Diagrama de secuencia configuración manual

El siguiente diagrama, es válido para 2 casuísticas posibles en lo que se refiere a la funcionalidad de la aplicación, el bloqueo de aplicaciones cuando se activa el Wifi, o el bloqueo de aplicaciones para cuando el dispositivo comienza a cargar la batería.

.

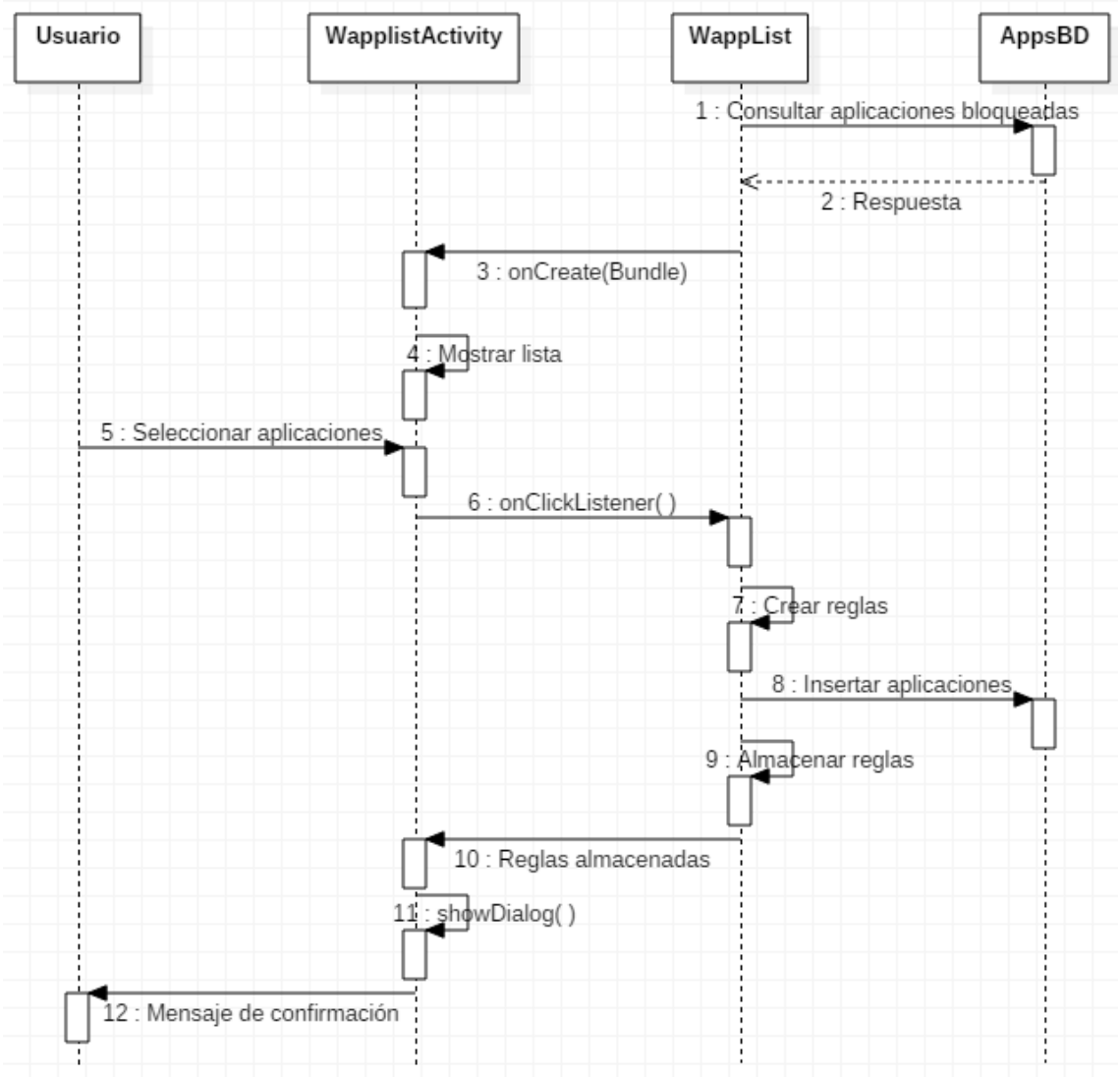


Ilustración 51. Diagrama de secuencia Configuración con Wifi/Batería

A continuación, se muestra el Diagrama de secuencia que se cumple en el caso de que un usuario desee modificar la contraseña de acceso a la aplicación.

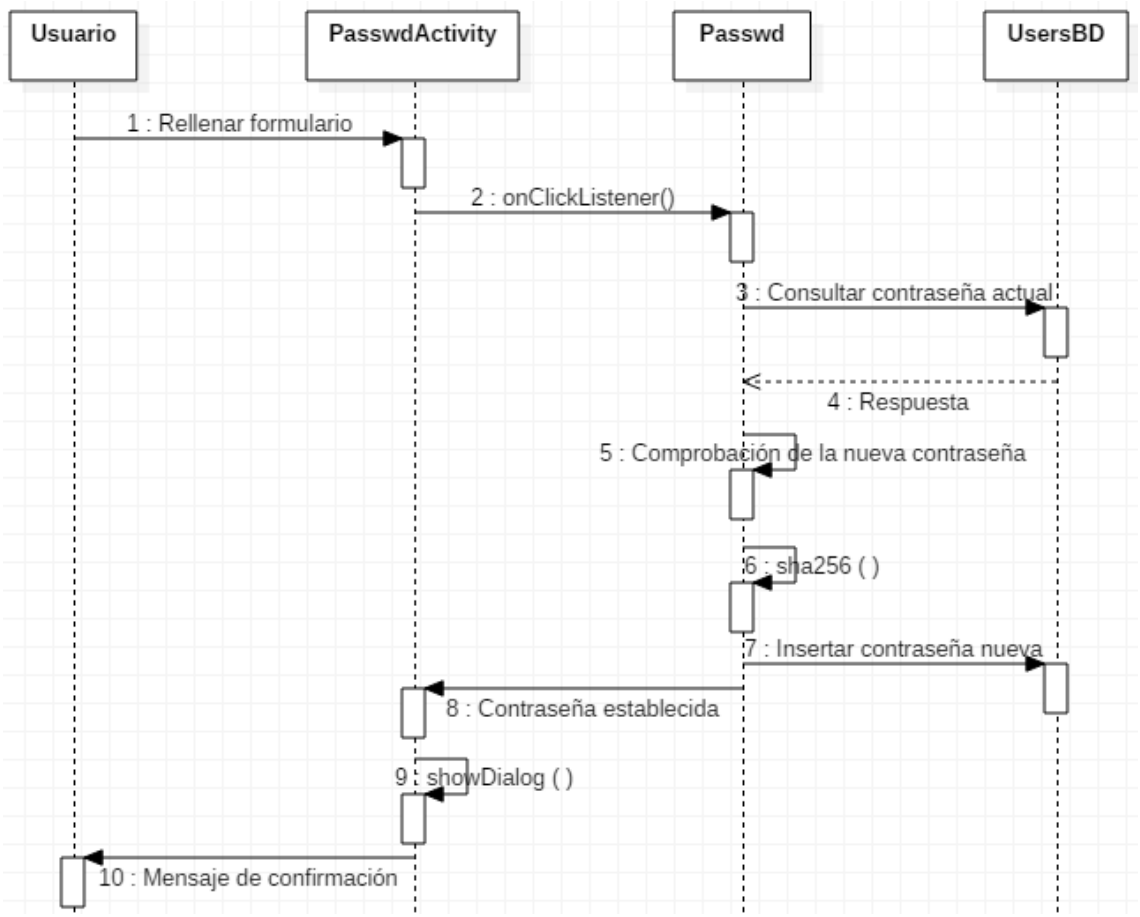


Ilustración 52. Diagrama de secuencia cambiar contraseña

En la Figura 53 se ilustra el diagrama de secuencia en el cual los componentes interactúan para activar (o desactivar) el mecanismo que gestiona la configuración automática de la aplicación. Este mecanismo se denomina Broadcast Receiver, y será detallado en el punto final de este capítulo.

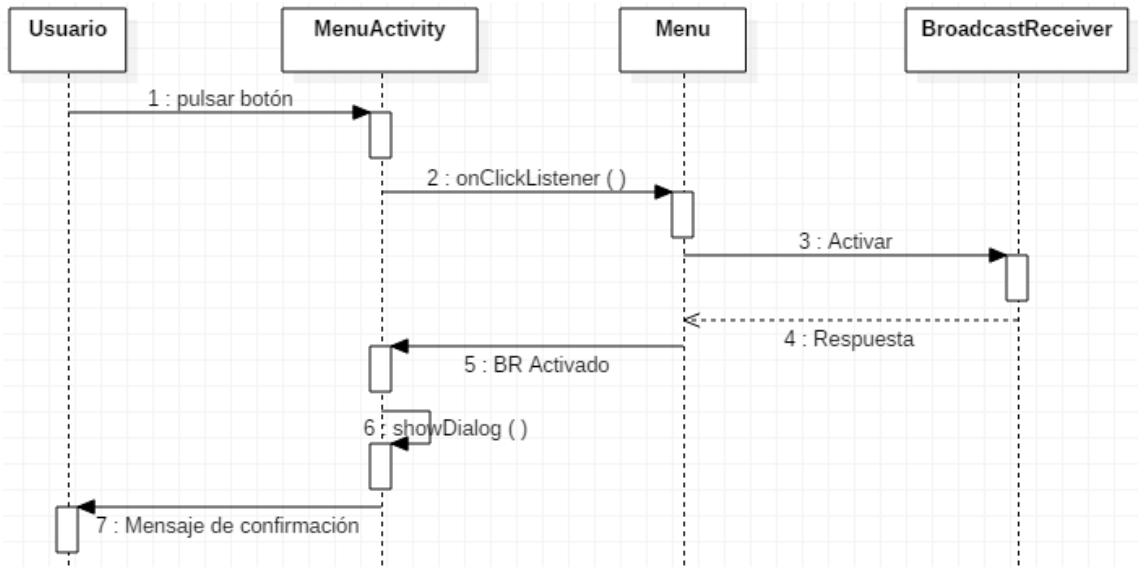


Ilustración 53. Diagrama de secuencia Configuración automática

Adicionalmente, se va a añadir un diagrama de secuencia más, el cual nos va a ilustrar sobre la interacción entre el Broadcast Receiver y los demás elementos de la aplicación cuando se encuentra activa a la configuración automática y se han de establecer las reglas de manera automática.

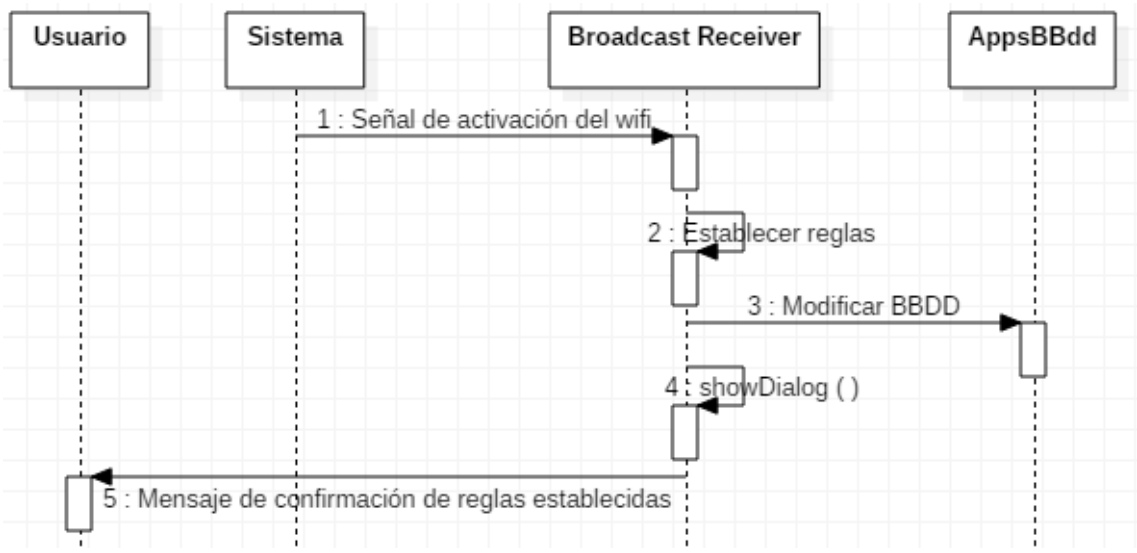


Ilustración 54. Diagrama de secuencia Establecimiento de Reglas automáticamente

4.3 Implementación

Una vez analizado el diseño, toca hablar de la implementación de la aplicación, en este punto se hablará y se justificarán ciertas decisiones de implementación que se han tomado así como algunos problemas que han surgido durante el desarrollo y las opciones que se han escogido para intentar solucionarlos.

4.3.1 Lenguaje, entorno y estructura

El lenguaje usado para el desarrollo de la aplicación, dejando a un lado las interfaces que se desarrollan usando XML, es el lenguaje Java, ya que es la única manera de desarrollar para este sistema operativo.

En lo que se refiere al entorno de desarrollo, se podrían usar varias herramientas que hay disponibles (NetBeans, Eclipse,etc) pero, finalmente, se ha decidido hacer uso de la aplicación Android Studio[14], ya que viene con una serie de características que hacen de este entorno de desarrollo una herramienta excelente para programar aplicaciones:

- Renderización en tiempo real.
- Consola de desarrollador,
- Consejos de optimización y ayuda para la traducción
- Soporte para la construcción de aplicaciones basada en Gradle
- Refactorización específica de Android y arreglos rápido..
- Uso de herramientas Lint para detectar problemas de rendimiento, usabilidad, etcétera.
- Plantillas para crear interfaces comunes de Android.
- Soporte para desarrollar aplicaciones para Android Wear.

[14] http://developer.android.com/intl/es/tools/studio/index.html

Ilustración 55. Android Studio

La aplicación de desarrollo Android Studio está disponible para Windows 2003, Windows Vista, Windows 8, Windows 10 y sistemas GNU/Linux y Mac OS X.

En la siguiente Figura se muestra la estructura del proyecto dentro del entorno de Android Studio.

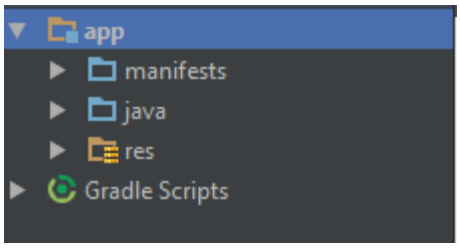


Ilustración 56. Estructura del proyecto

Tal y como se aprecia en la imagen, el proyecto consta de tres carpetas. A continuación se desglosa el contenido de cada una de las carpetas:

- **Manifest**: Esta carpeta contiene el fichero “Manifest.xml”, este archivo es un archivo de configuración donde se aplican las configuraciones básicas de la aplicación.

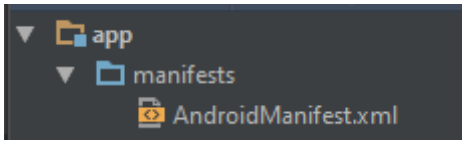


Ilustración 57. Carpeta manifest

- **Java**: Esta carpeta contiene los paquetes y las clases en Java que se encargan de proporcionar a la aplicación de una funcionalidad.

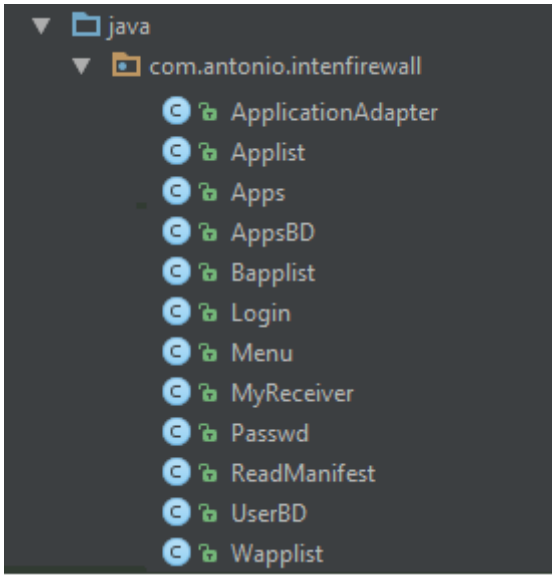


Ilustración 58. Carpeta Java

- **Res**: Esta carpeta contiene otras subcarpetas que a su vez contienen cosas como imágenes que se usan en la aplicación, los ficheros XML que crean las interfaces, iconos, etc.

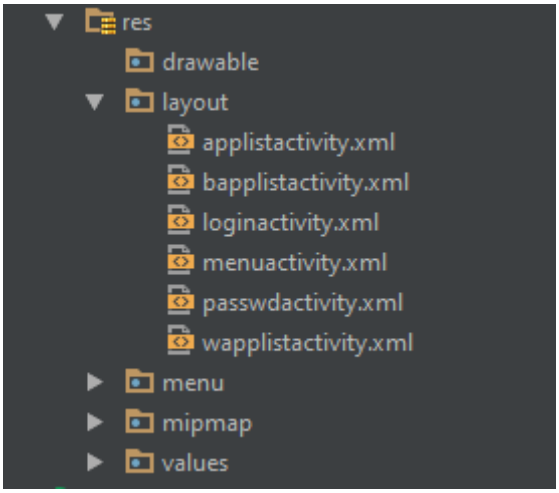


Ilustración 59. Carpeta Res

4.3.2Interfaz

La interfaz es uno de los puntos más importantes a tratar dentro del desarrollo de una aplicación. Dado que uno de requisitos de usuario exigía un diseño simple y minimalista, se ha optado por sobrecargar lo menos posible las interfaces de elementos decorativos.

Las interfaces en Android, reciben el nombre de “Actividades”. Dado que se ha optado por un diseño sencillo, la interfaz que se encarga de la acción de logarse o de establecer contraseña tendrá un aspecto como el descrito en la figura 60

Ilustración 60. Interfaz de Login

En el menú de la aplicación se ha optado por poner una serie de botones que dirija a cada una de las opciones que ofrece la aplicación. El texto de estos botones intenta detallar claramente a cuál de las funcionalidades de la aplicación apuntan.

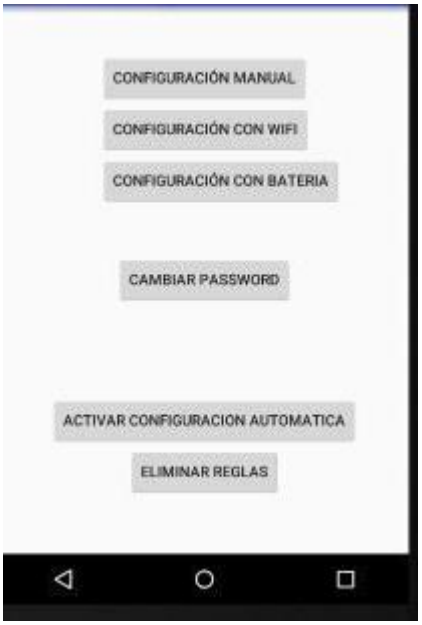


Ilustración 61. Interfaz del menú

Para representar las aplicaciones, se usa un objeto del tipo lista, tal y como se nos exigía en uno de los requisitos de usuario de la aplicación. Además, para realizar las acciones de bloqueo se ha establecido un botón que realizará todas las acciones pertinentes

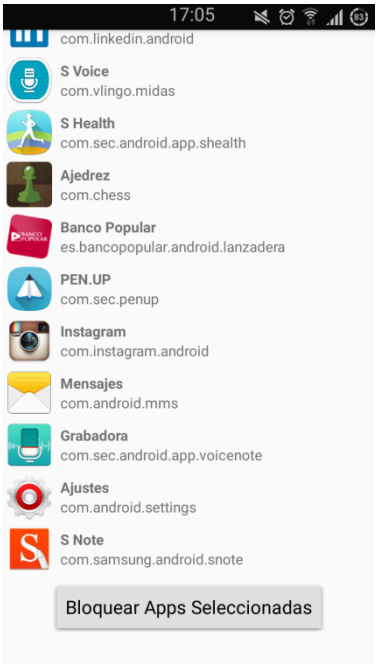


Ilustración 62. Interfaz para el bloqueo de aplicaciones

Para que el usuario pueda modificar la contraseña, se ha establecido un formulario donde el usuario deberá de introducir primeramente la contraseña actual, y a continuación deberá de introducir 2 veces (para evitar errores de escritura) la nueva contraseña que desea establecer.

4.3.3 Base de datos

Como ya se ha comentado en apartados anteriores, para el correcto funcionamiento de la aplicación se ha tenido que hacer uso de una Base de datos, con ella controlamos la contraseña de acceso a la aplicación, y mantenemos una coherencia y consistencia acerca del estado del cortafuegos para que el usuario pueda usarlo con mayor facilidad.

Para el uso de la Base de datos, Android dispone de un sistema de gestión denominado SQLite, este sistema es compatible con ACID (Atomicity, Consistency, Isolation and Durability) y posee bibliotecas escritas en C.

A diferencia de la mayoría de los sistemas gestores de bases de datos, SQLite no es un proceso independiente, en vez de eso se enlaza y se ejecuta de manera conjunta con la aplicación de la que forma parte. Gracias a esto se reduce la latencia del acceso a los datos.

En sus últimas versiones (Versión 3), SQLite [15]permite ya almacenar Bases de datos de hasta 2 Terabytes de capacidad.

Ilustración 63. Logo de SQLite

4.3.4 SHA256

Tal y como se especifica en los requisitos de usuario, la contraseña que da acceso a la aplicación debe de almacenarse protegida, de manera que si alguien tuviera acceso a la base de datos de la aplicación, no consiguiese obtener en claro el valor de la contraseña que nos permite acceder a la aplicación y manipular el cortafuegos.

Para proteger la contraseña se ha optado por hacer uso de una función hash o resumen, concretamente del algoritmo SAH256. Este algoritmo fue diseñado por la agencia de seguridad nacional (NSA).

Una función hash o resumen, es una función que transforma una cadena de bits de longitud indefinida, en una serie de caracteres de longitud fija. Esta transformación, aparte de no ser invertible, proporciona una salida de una cadena de caracteres lo más aleatoria posible.

En el caso de la función SHA256, se genera siempre como salida una cadena de 64 dígitos hexadecimales, tal y como se puede observar en la figura 64

Ilustración 64. Ejemplo SHA256

En este ejemplo se ha usado de contraseña la cadena “Password1234”, y al aplicarle el algoritmo obtenemos la cadena “a0f3285b07c26c0dcd2191447f391170

[15] https://www.sqlite.org/

d06035e8d57e31a048ba87074f3a9a15". Esta cadena es la que se almacena en la base datos y de este modo se impide que si alguien puede acceder a los datos almacenados en ella, se obtenga el valor en claro de la contraseña.

4.3.5 Permisos de Administrador

Para poder establecer las reglas en el interior del cortafuegos, es necesario que se disponga de permisos de administrador dentro del sistema, ya que el directorio donde se encuentra el cortafuegos no dispone de permisos de escritura para un usuario ordinario.

Es por ello que se convierte en característica imprescindible el que el dispositivo donde se ejecute la aplicación permita el login como administrador dentro del sistema. Para ello se puede hacer uso de distintas aplicaciones como "*SuperSU"*

Ilustración 65. Icono de SuperSU

A parte de ello, la aplicación debería de ser capaz de solicitar los permisos de administrador, para poder realizarlo, desde el código java, se ejecuta un comando "*su"* a nivel de sistema operativo, de esta manera se le concede a la aplicación durante un breve periodo de tiempo, permisos como administrador, permitiendo la interacción con todos los elementos del sistema operativo que requieren permisos de administrador.

4.3.6 Problemas encontrados

Como en todo proyecto, durante el desarrollo de la aplicación surgen ciertos problemas a los que hay que poner solución, a continuación se relatan los más importantes y como se ha procedido a su resolución:

- **Parseo de ficheros en XML**: Para que la aplicación pueda cumplir con su función, es necesario que pueda parsear el contenido de ficheros escritos en XML. Para ello se decidió por hacer uso de la interfaz *XML Pull Parser*, esta interfaz proviene de la API "XMLPULL V1" y nos permite recorrer de manera secuencial el fichero XML obteniendo cada uno de los valores que se almacenan dentro de las etiquetas del fichero. De esta manera se puede extraer el contenido que nos interesa del fichero y usarlo para nuestros propios fines.

- **Obtener Manifest**: Cuando un usuario desea bloquear una aplicación predeterminada, para que se puedan establecer correctamente las reglas, necesitamos tener conocimiento de las actividades que componen dicha aplicación. La única manera que tenemos de conocer las actividades de cada una de las aplicaciones es a través de su fichero Manifest, pero acceder a él desde dentro del sistema no es tan fácil.

  Cuando instalamos una aplicación en un dispositivo con Android, la aplicación se instala en forma de paquete, es decir, los elementos que hay en su interior (interfaces, clases en java, etc) no son accesibles por parte del usuario. Por lo tanto, para obtener el fichero Manifest, se ha optado por el desarrollo de una función que coge el paquete de la aplicación y lo recorre byte a byte hasta que encuentra el conjunto de bytes que conforman el fichero Manifest. A continuación a ese conjunto de bytes se les aplica una descompresión y de esta manera se van pasando ese conjunto de bytes a un conjunto de caracteres en XML que son legibles para la aplicación.

  De esta manera se reconstruye el Manifest de la aplicación y se pueden extraer sus actividades para generar las reglas.

- **Configuración automática:** Como ya se redactó en los requisitos, la aplicación debería de ser capaz de reconocer de manera automática ciertos eventos que tienen lugar en el dispositivo y, posteriormente, establecer una serie de reglas previamente determinadas por el usuario.

  Para solucionar este problema, se determinó la elaboración de un elemento que recibe el nombre de *Broadcast Receiver*[16], esta herramienta consiste en capturar ciertos eventos que tienen lugar en el sistema. Por ejemplo, si en el dispositivo se activa la conexión wireless, el dispositivo lanzará una señal a nivel de Sistema Operativo, y el *Broadcast Receiver* es capaz de recoger esa señal e interpretarla.

[16] http://developer.android.com/intl/es/reference/android/content/BroadcastReceiver.html

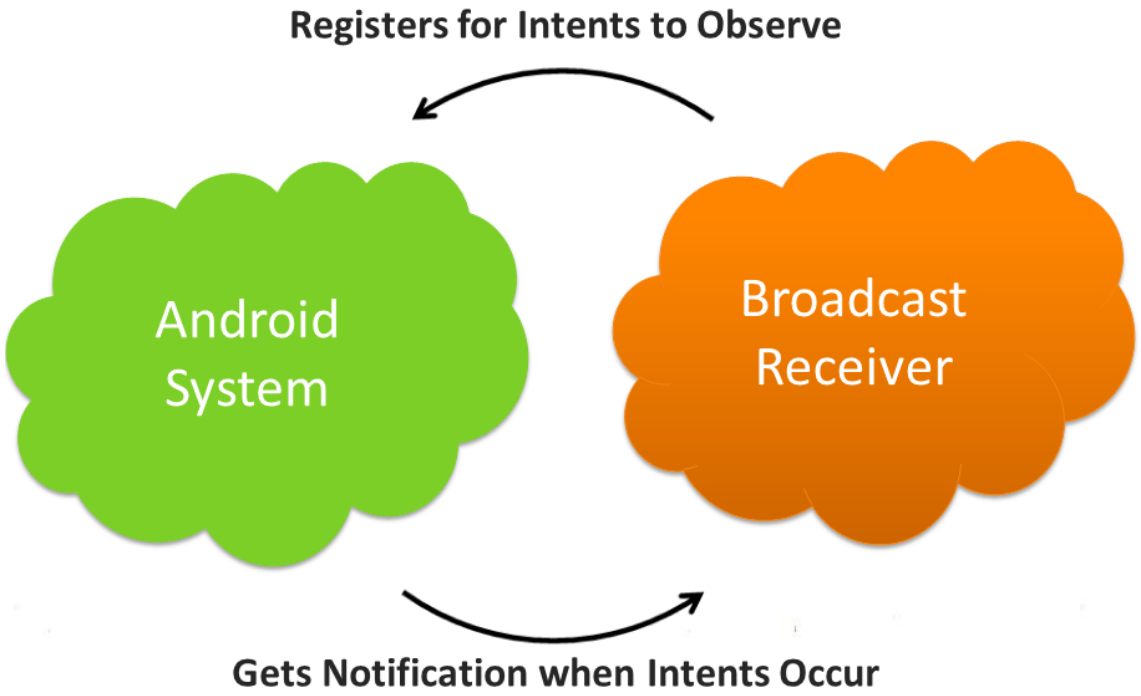


Ilustración 66. Funcionamiento de un Broadcast Receiver

De esta manera, cuando el *Broadcast Recevier* detecte algunos de los casos de configuración automática establecidos, él será el encargado de establecer las nuevas reglas dentro del cortafuegos.

CAPÍTULO 5

Pruebas y evaluación

En este capítulo se describen una serie de pruebas que se han llevado a cabo para garantizar que todos los aspectos que componen el sistema, especialmente los que hacen referencia a su comportamiento, se cumplen. Es decir, son pruebas de verificación de requisitos

Se describirá un entorno de pruebas, distintos tipos de pruebas y por cada prueba que se realice se representará con una tabla para facilitar su comprensión.

5.1 Descripción del entorno de pruebas

Para la elaboración de las pruebas se ha decidido emplear un dispositivo con sistema operativo Android y con una versión apta para el correcto funcionamiento del cortafuegos.

El dispositivo es un Samsung Galaxy Note II[17], este dispositivo posee 16 GG de memoria interna, 2 GB de memoria RAM y un procesador Exynos 4412 Quad con cuatro nucleos y 1,6GHz.

Tabla 61. Características del dispositivo de pruebas

Ilustración 67. Dispositivo de pruebas - Samsung Galaxy Note II

5.2 Lista de pruebas

Las pruebas, se dividen en 3 tipos: Pruebas unitarias, pruebas de integración y pruebas del sistema.

Las pruebas unitarias son las que determinan si cada uno de los métodos de un sistema funciona de la manera que debería. Para poder realizar estas pruebas, es necesario crear varias versiones pruebas por cada método, se le asigna un valor esperado al test y se ejecuta en modo testear. Si el resultado al finalizar la prueba es el mismo que

[17] http://www.smart-gsm.com/moviles/samsung-galaxy-note-2

el valor esperado, se concluye que los métodos funcionan correctamente. Debido a lo extensas que son estas pruebas, se ha optado por omitirlas en este documento.

Los test de integración se encargan de determinar que cada componente del sistema interactúa con los demás tal y como es debido. En este proyecto se llevaron a cabo varias pruebas de integración. Las pruebas fueron bastante sencillas, ya que bastaba con comprobar que las actividades interactuaban entre ellas y que la configuración automática se activaba y desactivaba cuando el usuario lo deseaba. No hacía falta testear más dado que no es necesario enviar a la aplicación ningún tipo de dato específico.

Finalmente quedan las pruebas del sistema, estas pruebas se encargan de testear funcionalidades concretas de la aplicación. A diferencia de los test unitarios, los cuales solo testean métodos, estos comprueban que los componentes se comportan como deberían. Puede darse la situación de que el método funcione correctamente, pero sin embargo, puede ser que el método no está haciendo lo que el cliente de verdad desea. Con estas pruebas se resuelven estos problemas.

A continuación se detallará un listado de pruebas del sistema que se han llevado a cabo. Para que sean más comprensibles las pruebas, se detallaran en unas tablas cuyos campos se completarán a medida que se vayan realizando las pruebas. De este modo las pruebas tendrán una estructura uniforme y se podrán planificar de mejor manera. Los campos que poseerán las tablas serán los siguientes:

- **ID:** Este campo representa la identificador único de la prueba. Este identificador estará compuesto por el código “PS – XX” donde XX será un número que empezará por el 01 y se irá correspondiendo con cada uno de los diagramas de secuencia establecidos en el apartado del diseño.

- **Descripción:** Este campo define el objetivo que se espera alcanzar con dicha prueba. La descripción será la que determine el resultado que se debe obtener para definir la prueba como superada o no. Por ejemplo, una salida esperada por parte de la aplicación sería un mensaje informativo informando que todo se ha realizado correctamente.

- **Pasos:** Este campo define y establece una serie de instrucciones que deben de realizarse para poder realizar la prueba.

- **Errores posibles:** Se trata de los errores que podrían suceder en el caso de que no se superase la prueba.

- **Requisitos:** Este campo hace referencia a los requisitos que se ven involucrados en las pruebas.

- **Estado:** Este campo determina el resultado de la prueba realizada, el campo podrá tener 2 valores “Superada” o “No superada”.

ID	PS - XX
Descripción	
Pasos	
Errores posibles	
Requisitos	
Estado	

Tabla 62. Plantilla de Pruebas del sistema.

A continuación se muestran todas las pruebas realizadas, las cuales se corresponden con los diagramas de secuencia realizados en el apartado de diseño de la aplicación.

ID	PS - 01
Descripción	**Establecer la contraseña de acceso a la aplicación**
Pasos	• **Abrir la aplicación** • **Introducir la contraseña** • **Pulsar el botón de “Establecer contraseña”**
Errores posibles	• **Que la aplicación no se abra** • **Que el campo de la contraseña esté vacío.**
Requisitos	• **RU – 07** • **RU – 12** • **RS – 04** • **RS - 09** • **RS - 18**
Estado	**Superada**

Tabla 63. PS – 01

ID	PS – 02
Descripción	**Autenticarse dentro de la aplicación**
Pasos	• **Abrir la aplicación** • **Introducir la contraseña** • **Pulsar el botón de "Acceder"**
Errores posibles	• **Que la aplicación no se abra** • **Que el campo de la contraseña este vacío.** • **Que la contraseña introducida no sea la correcta**
Requisitos	• **RU – 07** • **RU – 12** • **RS – 04** • **RS – 10** • **RS – 18**
Estado	**Superada**

Tabla 64. PS - 02

ID	PS – 03
Descripción	**Bloquear acceso a las aplicaciones de manera manual**
Pasos	• **Seleccionar en el menú el botón de "Configuración manual"** • **Seleccionar las aplicaciones de la lista** • **Pulsar sobre el botón "Bloquear Apps seleccionadas"**
Errores posibles	• **Que no se abra la actividad que nos lista las aplicaciones** • **Que no se carguen los elementos en la lista** • **Que no se soliciten permisos de administrador** • **Que no se establezcan correctamente las reglas** • **Que las aplicaciones no se bloqueen**
Requisitos	• **RU – 01** • **RU – 02** • **RU – 06** • **RU – 13** • **RU – 15** • **RU – 16** • **RS – 02** • **RS – 03** • **RS – 05** • **RS – 06** • **RS – 08** • **RS – 13** • **RS - 15**
Estado	**Superada**

Tabla 65. PS – 03

ID	PS – 04
Descripción	**Establece reglas para cuando se activa el Wifi/ O el dispositivo se encuentra cargando.**
Pasos	• **Seleccionar en el menú el botón de "Configuración con Wifi" o "Configuración con Batería"** • **Seleccionar las aplicaciones de la lista** • **Pulsar sobre el botón "Establecer reglas"**
Errores posibles	• **Que no se abra la actividad que nos lista las aplicaciones** • **Que no se carguen los elementos en la lista** • **Que no se generen correctamente las reglas**
Requisitos	• **RU – 01** • **RU – 02** • **RU – 04** • **RU – 05** • **RU – 10** • **RU – 13** • **RU – 16** • **RS – 02** • **RS – 03** • **RS – 05** • **RS – 06** • **RS – 07** • **RS – 08** • **RS – 14** • **RS- 15**
Estado	**Superada**

Tabla 66. PS – 04

ID	PS – 05
Descripción	**Modificar la contraseña de acceso a la aplicación**
Pasos	• **Seleccionar en el menú el botón de "Cambiar contraseña"** • **Introducir la contraseña actual** • **Introducir la contraseña nueva** • **Verificar la contraseña nueva** • **Pulsar sobre el botón "Establecer contraseña"**
Errores posibles	• **Que no se abra la actividad que nos muestra el formulario para introducir los datos** • **Que la contraseña actual sea errónea** • **Que la verificación de la contraseña sea errónea** • **Que la contraseña nueva esté vacía.** • **Que la contraseña no se cambie**
Requisitos	• **RU – 07** • **RU – 08** • **RU – 12** • **RS – 04** • **RS – 08** • **RS – 11** • **RS – 16** • **RS – 17** • **RS – 18**
Estado	**Superada**

Tabla 67. PS – 05

ID	PS – 06
Descripción	**Activar/Desactivar configuración automática**
Pasos	• **Seleccionar en el menú el botón de "Activar/Desactivar configuración automática"**
Errores posibles	• **Que no se active/desactive la configuración automática**
Requisitos	• **RU – 03** • **RU – 10** • **RU – 14** • **RS – 08** • **RS – 20**
Estado	**Superada**

Tabla 68. PS – 06

ID	PS – 07
Descripción	**Establecimiento de Reglas de manera automática**
Pasos	• **Tener unas reglas automáticas ya creadas** • **Activar la configuración automática**
Errores posibles	• **Que no se establezcan las reglas correctamente** • **Que no se bloqueen las aplicaciones**
Requisitos	• **RU – 03** • **RU – 04** • **RU – 05** • **RU – 10** • **RU – 14** • **RU – 15** • **RS – 01** • **RS – 05** • **RS – 08** • **RS – 13** • **RS – 14** • **RS – 19** • **RS – 20**
Estado	**Superada**

Tabla 69. PS – 07

Adicionalmente, se añaden dos pruebas más. En la primera, tras establecer unas reglas (ya sea de manera manual o automática) se verifica que las aplicaciones bloqueadas realmente se encuentran bloqueadas.

Mientras que en la segunda, se verifica si la aplicación borra correctamente las reglas y restablece el comportamiento del cortafuegos correctamente

ID	PS – 08
Descripción	**Comprobar que las aplicaciones están bloqueadas**
Pasos	• **Establecer unas reglas (ya sea de manera manual o automática)** • **Intentar acceder a una aplicación anotada en las reglas**
Errores posibles	• **Que la aplicación que supuestamente está bloqueada, se abra**
Requisitos	• **RU – 01** • **RU – 04** • **RU – 05** • **RU – 06** • **RU – 11** • **RS – 01** • **RS – 12** • **RS - 13**
Estado	**Superada**

Tabla 70. PS – 08

ID	PS – 09
Descripción	**Comprobar que las reglas se borran**
Pasos	• **Pulsar sobre el botón "Limpiar Reglas" del menú** • **Intentar acceder a una aplicación que estaba bloqueada anteriormente**
Errores posibles	• **Que la aplicación siga bloqueada**
Requisitos	• **RU – 09** • **RU – 11** • **RS – 05** • **RS – 08** • **RS – 12** • **RS – 13**
Estado	**Superada**

Tabla 71. PS - 09

5.2.1 Matriz de trazabilidad de pruebas

Para finalizar este apartado, se ha elaborado una tabla, donde se muestran todos los requisitos y todas las pruebas realizadas. De esta manera se demuestra que todos los requisitos establecidos en la fase de análisis han pasado las pruebas correctamente y, por lo tanto, cumplen con su función

	PS - 01	PS - 02	PS - 03	PS - 04	PS - 05	PS - 06	PS - 07	PS - 08	PS – 09
RU – 01			X	X				X	
RU – 02			X	X					
RU – 03						X	X		
RU – 04				X			X	X	
RU – 05				X			X	X	
RU – 06			X					X	
RU – 07	X	X			X				
RU – 08					X				
RU – 09									X
RU – 10				X		X	X		
RU – 11								X	X
RU – 12	X	X			X				
RU – 13			X	X					
RU – 14						X	X		
RU – 15			X				X		
RU – 16			X	X					
RS – 01							X	X	
RS – 02			X	X					
RS – 03			X	X					
RS – 04	X	X			X				
RS – 05			X	X			X		X
RS – 06			X	X					
RS – 07				X					
RS – 08			X	X	X	X	X		X
RS – 09	X								
RS – 10		X							
RS – 11					X				
RS – 12								X	X
RS – 13			X				X	X	X
RS – 14				X			X		
RS – 15			X	X					
RS – 16					X				
RS – 17					X				
RS – 18	X	X			X				
RS – 19							X		
RS – 20						X	X		

Tabla 72. Trazabilidad de requisitos y pruebas

CAPÍTULO 6

Planificación y Presupuesto

En este apartado se muestra la planificación que se ha ido realizando durante el desarrollo de la aplicación. Para ello se hará uso de un diagrama de Gantt. Además se mostrará un informe donde se incluirán todos los detalles para el establecimiento de un presupuesto que nos indique el coste de la aplicación.

6.1 Planificación

A la hora de realizar la imputación de horas, se ha decidido por hacer primero una imputación inicial que se hizo al inicio del proyecto, y luego una imputación final con los datos reales de las horas que nos ha llevado confeccionar el proyecto. De esta manera se puede realizar una comparación de los datos para ver si ha habido un gran desvío en relación con la imputación inicial.

En la siguiente ilustración se muestra el recurso del proyecto y las distintas tareas asignadas

Nombre	Función
Antonio Requena López	Encargado del proyecto
Iniciación con Android	
Análisis del problema	
Redacción de requisitos	
Casos de uso	
Arquitectura de la aplicación	
Desarrollo de Diagramas	
Desarrollo de la Interfaz	
Desarrollo de la lógica de negocio	
Tests	
Elaboración de la memoria	

Ilustración 68. Etapas de trabajo

Estas tareas están agrupadas en 4 grupos distintos:

- Análisis: Etapa en la que se identifica el problema y se redactan los requisitos.
- Diseño: Etapa donde se definen los casos de uso, la arquitectura y los diagramas de secuencia.
- Implementación: Etapa en la que se desarrolla la aplicación
- Elaboración de la memoria: Elaboración del presente documento donde se reflejan cada uno de los apartados y el tiempo que han llevado su elaboración.

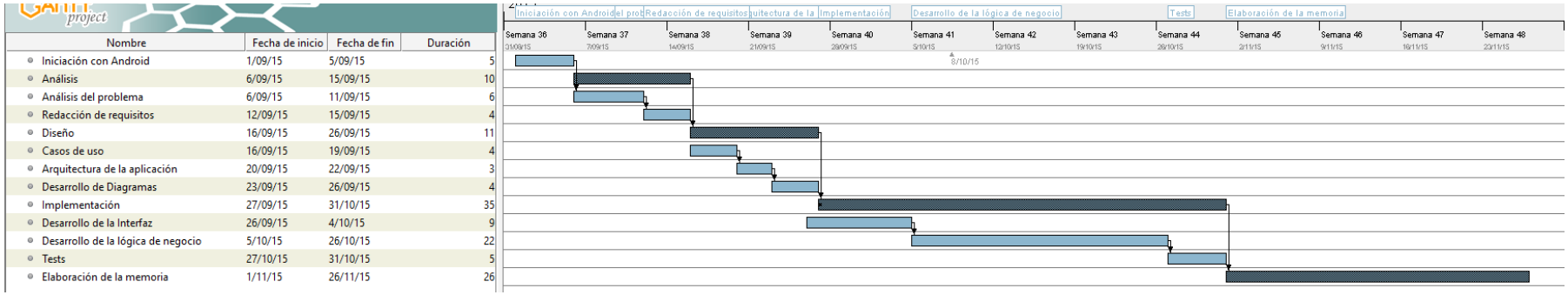
En la ilustración 3 se muestra una planificación inicial de cada una de las etapas del trabajo. A continuación, en la Ilustración 4 se muestra la planificación final, con el tiempo real que llevó elaborar cada uno de los apartados, de esta manera se puede apreciar que hay una desviación en el tiempo estimado de unos 60 días.

Iniciación con Android	1/09/15	5/09/15	5
Análisis	6/09/15	15/09/15	10
Análisis del problema	6/09/15	11/09/15	6
Redacción de requisitos	12/09/15	15/09/15	4
Diseño	16/09/15	26/09/15	11
Casos de uso	16/09/15	19/09/15	4
Arquitectura de la aplicación	20/09/15	22/09/15	3
Desarrollo de Diagramas	23/09/15	26/09/15	4
Implementación	27/09/15	31/10/15	35
Desarrollo de la Interfaz	26/09/15	4/10/15	9
Desarrollo de la lógica de negocio	5/10/15	26/10/15	22
Tests	27/10/15	31/10/15	5
Elaboración de la memoria	1/11/15	26/11/15	26

Ilustración 69. Planificación Inicial

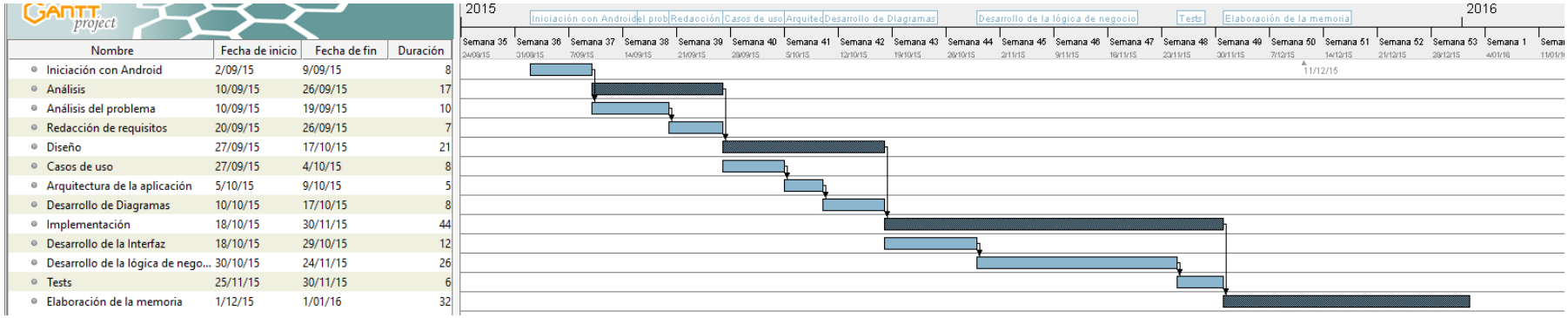
Nombre	Fecha de inicio	Fecha de fin	Duración
Iniciación con Android	2/09/15	9/09/15	8
Análisis	10/09/15	26/09/15	17
Análisis del problema	10/09/15	19/09/15	10
Redacción de requisitos	20/09/15	26/09/15	7
Diseño	27/09/15	17/10/15	21
Casos de uso	27/09/15	4/10/15	8
Arquitectura de la aplicación	5/10/15	9/10/15	5
Desarrollo de Diagramas	10/10/15	17/10/15	8
Implementación	18/10/15	30/11/15	44
Desarrollo de la Interfaz	18/10/15	29/10/15	12
Desarrollo de la lógica de nego...	30/10/15	24/11/15	26
Tests	25/11/15	30/11/15	6
Elaboración de la memoria	1/12/15	1/01/16	32

Ilustración 70. Planificación Final



Nombre	Fecha de inicio	Fecha de fin	Duración
Iniciación con Android	1/09/15	5/09/15	5
Análisis	6/09/15	15/09/15	10
Análisis del problema	6/09/15	11/09/15	6
Redacción de requisitos	12/09/15	15/09/15	4
Diseño	16/09/15	26/09/15	11
Casos de uso	16/09/15	19/09/15	4
Arquitectura de la aplicación	20/09/15	22/09/15	3
Desarrollo de Diagramas	23/09/15	26/09/15	4
Implementación	27/09/15	31/10/15	35
Desarrollo de la Interfaz	26/09/15	4/10/15	9
Desarrollo de la lógica de negocio	5/10/15	26/10/15	22
Tests	27/10/15	31/10/15	5
Elaboración de la memoria	1/11/15	26/11/15	26

Ilustración 71. Diagrama de Gantt Inicial



Nombre	Fecha de inicio	Fecha de fin	Duración
Iniciación con Android	2/09/15	9/09/15	8
Análisis	10/09/15	26/09/15	17
Análisis del problema	10/09/15	19/09/15	10
Redacción de requisitos	20/09/15	26/09/15	7
Diseño	27/09/15	17/10/15	21
Casos de uso	27/09/15	4/10/15	8
Arquitectura de la aplicación	5/10/15	9/10/15	5
Desarrollo de Diagramas	10/10/15	17/10/15	8
Implementación	18/10/15	30/11/15	44
Desarrollo de la Interfaz	18/10/15	29/10/15	12
Desarrollo de la lógica de nego...	30/10/15	24/11/15	26
Tests	25/11/15	30/11/15	6
Elaboración de la memoria	1/12/15	1/01/16	32

Ilustración 72. Diagrama de Gantt Final

6.2 Presupuesto

A continuación se muestra el presupuesto que ha implicado el desarrollo de la aplicación, describiendo gastos en material y en personal. El presupuesto total de este proyecto asciende a la cantidad de **9.287,185 €.**

En el presupuesto no se contemplan costes de licencias de software, ya que todo el software usado durante el desarrollo es de código abierto, es decir, es gratis.

En las tablas del presupuesto se muestran datos como los nombres y los salarios de las personas que han contribuido a la elaboración del proyecto, así como el tiempo dedicado y el material usado y sus costes.

UNIVERSIDAD CARLOS III DE MADRID
Escuela Politécnica Superior

PRESUPUESTO DE PROYECTO

1.- Autor:
Antonio Requena López

2.- Departamento:
UC3M Computer security Lab

3.- Descripción del Proyecto:

- Titulo	**Diseño e implementación de una aplicación para la gestión del cortafuegos de Android**
- Duración (meses)	**4**
Tasa de costes Indirectos:	**20%**

4.- Presupuesto total del Proyecto (valores en Euros):
10.000,00 Euros

5.- Desglose presupuestario (costes directos)

PERSONAL

Apellidos y nombre	Categoría	Dedicación (hombres mes)	Coste hombre mes	Coste (Euro)
Sergio Pastrana Portillo	Ingeniero Senior	370	4.289,54	4.160.853,80
Antonio requena López	Ingeniero	1325	2.694,39	3.570.066,75
			Total	7.730.920,55

[4] 1 Hombre mes = 131,25 horas. Máximo anual de dedicación de 12 hombres mes (1575 horas)
Máximo anual para PDI de la Universidad Carlos III de Madrid de 8,8 hombres mes (1.155 horas)

EQUIPOS

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable [4]
Packard Bell imedia s1800	549,99	100	4	60	36,67
Samsung Galaxy Note II	459,99	100	4	60	30,67
				Total	67,33

[4] Fórmula de cálculo de la Amortización:

$$\frac{A}{B} x C x D$$

A = nº de meses desde la fecha de facturación en que el equipo es utilizado
B = periodo de depreciación (60 meses)
C = coste del equipo (sin IVA)
D = % del uso que se dedica al proyecto (habitualmente 100%)

6.- Resumen de costes

Presupuesto Costes Totales	**Presupuesto Costes Totales**
Personal	7.730.921
Amortización	67
Costes Indirectos	1.546.198
Total	9.277.185

Ilustración 73. Presupuesto del proyecto

CAPÍTULO 7

Conclusiones

En este apartado se recogen una serie de conclusiones a las cuales se han llegado tras la elaboración de este proyecto

En este Trabajo de Fin de Grado se ha presentado el análisis, diseño e implementación de una aplicación funcional que permite al usuario configurar el bloqueo de aplicaciones en determinadas circunstancias. Primero, se ha realizado un estudio acerca de cuáles son las herramientas actuales que pudieran utilizarse, concluyendo que Android posee un cortafuegos interno que pudo servir de base para este proyecto. Fue necesaria también la profundización en la arquitectura del SO Android, ya que para poder bloquear las aplicaciones era necesario conocer las actividades que las componen.

El sistema completo está elaborado sobre el patrón arquitectónico del Modelo Vista Controlador, también denominado MVC, permitiendo separar la lógica del sistema de las interfaces y del acceso a los datos. Esto resulta bastante útil para fomentar la extensibilidad y reutilización de un sistema que está construido de manera modular y podrá albergar en un futuro nuevas funciones para que garantizar un servicio más completo de cara al usuario.

Gracias a las pruebas del sistema que se han realizado, queda demostrado que los requisitos redactados en la parte de análisis se cumplen, cumpliendo de esta manera con las expectativas del cliente.

Durante la elaboración de este proyecto se han adquirido una serie de conocimientos a la par que se profundizaba en otros de los cuales ya se tenía conocimiento. Si bien ha sido necesario aprender a programar en Android prácticamente desde cero, ese no ha sido el punto en el que más se ha necesitado profundizar. Dado que el funcionamiento del cortafuegos de Android no dispone de documentación oficial por parte de Google fueron necesarias muchas horas de pruebas y de búsqueda de información para entender su funcionamiento, su configuración y todas las posibilidades que este ofrece.

Durante el desarrollo surgieron ciertos problemas a la hora de programar, pero consultando portales en internet se subsanaron sin suponer un percance grave, aunque el tiempo estimado de la etapa de programación de la aplicación se vio aumentado en unos días.

Finalmente, recalcar que se produjo una desviación en el tiempo estimado inicialmente para la elaboración del proyecto, ya que durante el desarrollo y la investigación inicial fue necesario invertir más tiempo del previsto inicialmente. Del mismo modo, durante el desarrollo del presente documento, fue necesaria recabar información de manera más exhaustiva y veraz para algunos apartado (Como el de Estado del Arte) lo cual desembocó en una inversión extra de horas a parte de las ya previstas.

En lo referente al aprendizaje personal, este trabajo ha supuesto todo un reto en todas y cada una de sus fases, ya que era la primera vez que se hacía frente a un proyecto de semejantes magnitudes. El tener que elaborar un modelo arquitectónico completo, y adquirir ciertos conocimientos a través de la búsqueda y de la constatación

de información en diferentes portales de internet, fue una tarea dura en la que se tuvo que emplear una gran cantidad de horas, aunque el esfuerzo se vio recompensado con el resultado final de la aplicación

Este proyecto ha servido para aprender el funcionamiento de una tecnología actual y con una presencia en el mercado importante como es Android, a la par de servir de simulación de un trabajo real que podría exigirse perfectamente fuera del ámbito académico en el que se han de cumplir unas pautas y unas exigencias por parte de un cliente

CAPÍTULO 7

Anexos

En este apartado se recogen una serie de términos y de abreviaturas incluidas en el documento, que permitan al lector entender los conceptos a los que se hace referencia

7.1 Terminología

En este apartado se incluyen todos aquellos términos y abreviaturas que se consideran necesarios definir para la total comprensión de este documento.

7.1.1 Glosario de términos

Términos	Descripción
Smartphone	También denominados teléfonos inteligentes, son dispositivos que poseen un sistema operativo que les permite adaptarse a su dueño y a las circunstancias del mismo
Android	Es un sistema operativo creado por Google para los smartphones.
iOS	Sistema operativo creado por Apple para sus smartphones, los iPhones
Windows Phone	Sistema operativo para smartphones desarrollado por Microsoft
BlackBerry OS	Sistema operativo para smartphones desarrollado por la BackBerry
Play Store	Es el portal oficial de aplicaciones de la plataforma Android, creado por Google.
Wifi	Mecanismo de conexión inalámbrica entre dispositivos. Se usa habitualmente para conectarse a internet de manera inalámbrica
Google	Empresa multinacional americana basada en proporcionar todo tipo de servicios online
Cortafuegos	Parte de un sistema para bloquear un acceso no autorizado, permitiendo al mismo tiempo las comunicaciones autorizadas.
Android Studio	Entorno de desarrollo oficial integrado para la plataforma Android
Java	Lenguaje de programación orientado a objetos.
Función Hash	También conocida como función Resumen. Es una función computable mediante un algoritmo, se usa para proteger caracteres de manera que no se almacenan en texto plano.

Tabla 73. Glosario de términos

7.1.2 Abreviaturas

Abreviaturas	Descripción
API	Application Programming Interface. Es el conjunto de funciones con que se provee al programador para interactuar con un determinado sistema
ADB	Android Debug Bridge, es una herramienta que conecta un dispositivo Android con un entorno de desarrollo.
SDK	Software Development Kit, es una herramienta que proporciona al desarrollador todas las librerías necesarias para poder trabajar con un determinado lenguaje
XML	eXtended Marckup Language, se trata de un lenguaje que permite

	crear mensajes e intercambiarlo entre plataformas.
UML	Unified Modeling Language. Lenguaje de modelado de sistemas de software.
MVC	Modelo – Vista - Controlador
SQL	Structured Query Language. Es un lenguaje declarativo de acceso a bases de datos

Tabla 74. Glosario de abreviaturas